(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 3 220 274 A1

(12) **EUROPEAN PATENT APPLICATION**
published in accordance with Art. 153(4) EPC

(43) Date of publication:
20.09.2017 Bulletin 2017/38

(21) Application number: **15870204.3**

(22) Date of filing: **17.11.2015**

(51) Int Cl.:
G06F 12/02 *(2006.01)*

(86) International application number:
PCT/KR2015/012317

(87) International publication number:
WO 2016/099036 (23.06.2016 Gazette 2016/25)

(84) Designated Contracting States:
AL AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HR HU IE IS IT LI LT LU LV MC MK MT NL NO PL PT RO RS SE SI SK SM TR
Designated Extension States:
BA ME
Designated Validation States:
MA MD

(30) Priority: **15.12.2014 KR 20140180501**

(71) Applicant: **Samsung Electronics Co., Ltd.**
Gyeonggi-do 16677 (KR)

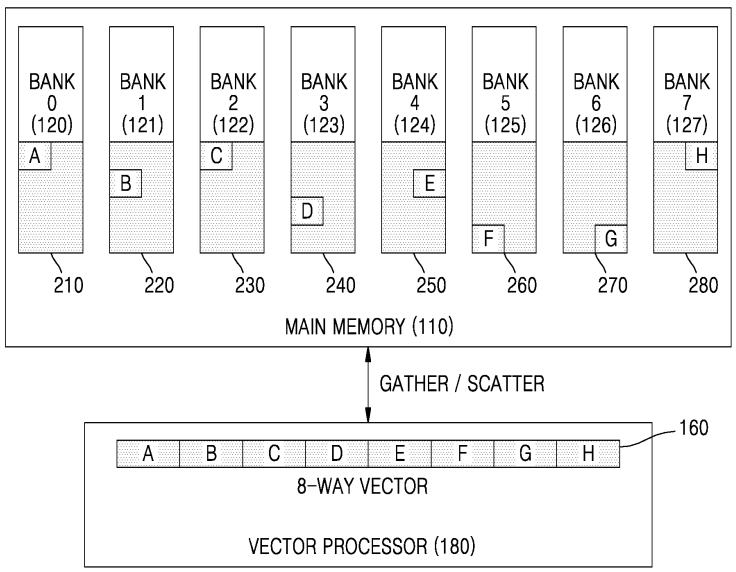
(72) Inventors:
- **CHO, Jung-uk**
  Hwaseong-si
  Gyeonggi-do 18430 (KR)
- **KIM, Suk-jin**
  Seoul 04161 (KR)
- **SUH, Dong-kwan**
  Yongin-si
  Gyeonggi-do 16951 (KR)

(74) Representative: **Appleyard Lees IP LLP**
15 Clare Road
Halifax HX1 2HY (GB)

(54) **METHOD AND APPARATUS FOR MEMORY ACCESS**

(57) Disclosed is an apparatus comprising: a plurality of memory banks; and a controller for generating a plurality of lookup tables storing data, needed for vector arithmetic operations, copied from data stored in the plurality of memory banks, and generating vector data by reading the data in the generated lookup tables.

FIG. 2



EP 3 220 274 A1

Description

TECHNICAL FIELD

[0001] The inventive concept relates to a method and apparatus by which a processor accesses a memory, and more particularly, to a method and apparatus by which a vector processor gathers a plurality of pieces of data from a memory and scatters the plurality of pieces of data back in the memory.

BACKGROUND ART

[0002] A processor may access a memory to read data that is necessary for an operation and to store results of the operation back in the memory. For example, a processor may execute a load instruction that reads data from a memory and a store instruction that stores data in the memory.

[0003] In early processor technologies, scalar processors which employ a single instruction single data (SISD) method, which is a method of processing a single piece of data with a single instruction, have been used.

[0004] However, with the spread of smart phones and high-definition televisions, the necessity for processors capable of processing a large amount of data in fields such as image processing, vision processing, image quality processing, and graphic rendering has increased. Accordingly, vector processors which employ a single instruction multiple data (SIMD) method, which is a method of processing a plurality of pieces of data with a single instruction, have been generalized. A vector processor is a processor for repeatedly performing the same operation at a high speed on a plurality of pieces of data constituting a vector. The vector processor may simultaneously read and process a plurality of pieces of data from a memory and then store a result thereof back in the memory.

[0005] When a vector processor simultaneously reads a plurality of pieces of data from a memory or stores the plurality of pieces of data in the memory, the vector processor frequently accesses the same memory bank. In this case, a memory bank conflict causes a stall, and thus performance of a system is degraded.

[0006] Therefore, in order for the vector processor to simultaneously read the plurality of pieces of data from the memory or to store the plurality of pieces of data in the memory, dedicated hardware may be additionally required to predict memory bank conflicts and to minimize the occurrence of the conflictions by scheduling memory accesses of the vector processor based on the prediction.

[0007] However, the additional hardware may increase an overall area of the processor and increase design complexity. Also, complex implementation in software may cause another performance degradation of the processor.

TECHNICAL PROBLEM

[0008] The inventive concept provides a method and apparatus by which a processor reads a plurality of pieces of data in the form of a lookup table from a memory or stores the plurality of pieces of data in the form of a lookup table in the memory.

[0009] Further, the inventive concept provides a computer readable recording medium in which a program for causing a computer to execute the method is recorded. Technical objectives of embodiments of the inventive concept are not limited to the above objectives, and other objectives can be deduced from the following embodiments.

BEST MODE

DESCRIPTION OF THE DRAWINGS

[0010]

FIG. 1 is a diagram illustrating a structure in which a vector processor according to one embodiment reads and writes data from and to a memory.

FIG. 2 is a diagram illustrating a method by which a vector processor according to one embodiment performs a gather and a scatter.

FIG. 3 is a diagram illustrating a structure of a vector processor according to one embodiment.

FIG. 4 is a diagram illustrating a plurality of lookup tables generated in a main memory according to one embodiment.

FIG. 5 is a diagram illustrating a plurality of lookup tables generated in a main memory according to one embodiment.

FIG. 6A is a diagram illustrating a structure of a vector processor according to one embodiment.

FIG. 6B is a diagram illustrating a method by which a vector processor according to one embodiment synchronizes a plurality of lookup tables.

FIG. 7 is a diagram illustrating a structure of a vector processor according to one embodiment.

FIG. 8A is a diagram illustrating a method by which a vector processor according to one embodiment performs a gather.

FIG. 8B is a diagram illustrating a method by which a vector processor according to one embodiment performs a gather.

FIG. 9 is a diagram illustrating a method by which a vector processor according to one embodiment performs a scatter.

FIG. 10 is a flowchart illustrating a method by which a vector processor according to one embodiment performs a gather.

FIG. 11 is a flowchart illustrating a method by which a vector processor according to one embodiment generates a plurality of lookup tables.

FIG. 12 is a flowchart illustrating a method by which a vector processor according to one embodiment performs a scatter.

FIG. 13 is a flowchart illustrating a method by which a vector processor according to one embodiment performs a gather and a scatter.

TECHNICAL SOLUTION

[0011] According to an aspect of the inventive concept, there is provided an apparatus including a plurality of memory banks, and a controller configured to generate a plurality of lookup tables in which data that is necessary for a vector operation among data stored in the plurality of memory banks is copied and stored, and to generate vector data by reading the data from the plurality of lookup tables.

[0012] The controller may include a lookup table generating unit configured to generate the plurality of lookup tables, and a gather unit configured to read the data from the plurality of lookup tables and generate the vector data.

[0013] The controller may generate one lookup table for each of the plurality of memory banks, wherein a number of the generated lookup table is same as the number of the plurality of memory banks.

[0014] The controller may divide the plurality of memory banks into a predetermined number of groups and generate one lookup table for each of the groups, and the one lookup table generated for each of the groups may be stored in a plurality of memory banks included in the group in an interleaving form.

[0015] The controller may access each of the lookup tables using an index vector including a plurality of randomly generated indexes and read data stored at positions of the indexes in each of the lookup tables.

[0016] The controller may divide result vector data obtained by performing a predetermined vector operation on the vector data into elements and store the result vector divided by elements in the plurality of lookup tables.

[0017] The controller may store each of the elements at a predetermined index position in each of the lookup tables.

[0018] When a value of data stored in a first index in any one of the plurality of lookup tables is changed, the controller may update data stored in a first index in each of the remaining lookup tables, whose value is not changed, with the changed value.

[0019] The apparatus may further include a switching unit including a plurality of sub-switch units corresponding to the lookup tables, wherein the plurality of sub-switch units may determine whether to allow the controller to access a predetermined index position of each of the lookup tables.

[0020] The plurality of sub-switch units may include a plurality of switches corresponding to each of the plurality of memory banks, and each of the switches may determine whether the controller is accessible the each of the plurality of memory banks.

[0021] According to another aspect of the inventive concept, there is provided a method including generating a plurality of lookup tables in which data that is necessary for a vector operation among data stored in a plurality of memory banks is copied and stored, and generating vector data by reading the data from the lookup tables.

[0022] The generating of the lookup tables may include generating one lookup table for each of the plurality of memory banks, wherein a number of the generated lookup table is same as the number of the plurality of memory banks in each memory banks.

[0023] The generating of the lookup tables may include dividing the plurality of memory banks into a predetermined number of groups and generating one lookup table for each of the groups, and storing the one lookup table generated for each of the groups in a plurality of memory banks included in the group in an interleaving form.

[0024] The generating of the vector data by reading the data from the lookup tables may include accessing each of the lookup tables using an index vector including a plurality of randomly generated indexes and reading data stored at positions of the indexes in each of the lookup tables.

[0025] The method may further include dividing result vector data obtained by performing a predetermined vector operation on the vector data into elements and storing the result vector divided by elements in the plurality of lookup tables.

[0026] The storing of the result vector divided by elements in the plurality of lookup tables may include storing each of the elements at a predetermined index position in each of the lookup tables.

[0027] The storing of the result vector divided by elements in the plurality of lookup tables may include, when a value

of data stored in a first index in any one of the plurality of lookup tables is changed, updating data stored in a first index in each of the remaining lookup tables, whose value is not changed, with the changed value.

[0028] The generating of the vector data by reading the data from the lookup tables may include generating the vector data by reading data present at a predetermined index position that is allowed to be accessed in each of the lookup tables, and the storing of the result vector divided by elements in the plurality of lookup tables may include storing the result vector data at the predetermined index position that is allowed to be accessed in each of the lookup tables.

[0029] The generating of the vector data by reading the data from the lookup table and the storing of the result vector divided by elements in the plurality of lookup tables may include determining whether to allow access to each of the memory banks.

[0030] According to still another aspect of the inventive concept, there is provided a computer readable recording medium in which a program for causing a computer to execute the method is recorded.

DETAILED DESCRIPTION OF THE INVENTION

[0031] Hereinafter, embodiments will be described in detail with reference to the drawings. Since the embodiments described in this specification and configurations illustrated in the drawings are only exemplary embodiments of the inventive concept and do not represent the overall technological scope of the inventive concept, it should be understood that the inventive concept covers various equivalents, modifications, and substitutions at the time of filing of this application.

[0032] FIG. 1 is a diagram illustrating a structure in which a vector processor according to one embodiment reads and writes data from and to a memory.

[0033] A main memory 110 may be a random access memory (RAM) that constitutes a computer, a TV, a mobile phone, a mobile device, and the like, but the inventive concept is not limited thereto.

[0034] Hereinafter, the term "main memory" refers to all or some areas of a memory constituting a memory device of a predetermined machine. The main memory 110 according to one embodiment may have a multi-bank structure including one or more memory banks (hereinafter referred to as "banks") to minimize memory conflicts.

[0035] A vector processor 180 may be a processor capable of simultaneously processing a plurality of pieces of data, and instructions in the vector processor 180 may be processed in the form of single instruction multiple data (SIMD) or multiple instruction multiple data (MIMD), but the inventive concept is not limited thereto.

[0036] The vector processor 180 may read a plurality of pieces of data from the main memory 110, generate a plurality of pieces of data in the form of a vector, perform a vector operation thereon, and store a result of the vector operation back in the main memory 110. The vector processor 180 may access a plurality of banks 120, 121, 122, 123, 124, 125, 126, and 127 in the main memory 110 to read or store the plurality of pieces of data simultaneously.

[0037] The vector processor 180 may include a vector register 160. The vector register 160 is a register which may store an address of a memory for accessing elements constituting a vector operation and may read or write through a vector instruction.

[0038] The vector register 160 may be partitioned into a plurality of elements. For example, when a 16-byte vector register 160 stores an 8-way vector, the vector register 160 may be composed of eight elements and one element may have a size of 2 bytes.

[0039] Although not illustrated in FIG. 1, the vector processor 180 may include a vector functional unit for performing an operation every clock with a pipelined architecture, a vector load/store unit for reading and storing data from a memory, a scalar register for storing a memory address and a control signal, and a cross-bar for connecting registers.

[0040] The vector processor 180 according to one embodiment may use various methods to access the main memory 110 in order to read a plurality of pieces of data from the main memory 110 or to write a plurality of pieces of data to the main memory 110.

[0041] For example, the vector processor 180 may sequentially read a plurality of pieces of data from the main memory 110 while incrementing a memory address by one, and generate the plurality of pieces of data in the form of a vector. The vector processor 180 may use a unit stride technique that performs a vector operation using the generated vector and sequentially stores results of the performed vector operation back in the main memory 110.

[0042] Also, for example, the vector processor 180 may read a plurality of pieces of data from the main memory 110 while incrementing a memory address by a constant value stored in a scalar register (not illustrated), and generate the plurality of pieces of data in the form of a vector. The vector processor 180 may use a stride access technique that performs a vector operation using the generated vector and stores results obtained by performing the vector operation back in the main memory 110 while incrementing the memory address by the constant value in the same method.

[0043] Also, for example, the vector processor 180 may randomly read data from the main memory 110 and generate a plurality of pieces of the read data in the form of a vector. The vector processor 180 may use an indexed load and store technique that performs a vector operation using the generated vector and randomly stores results obtained by performing the vector operation back in the main memory 110. The indexed load and store technique is a technique of

reading scattered data in the main memory 110 using an index vector to generate a vector and storing results of the vector operation back in the main memory 110.

[0044] Generally, reading a plurality of pieces of data scattered in the main memory 110 using an index vector is referred to as a gather, and scattering and storing a plurality of pieces of data (i.e., a vector) in the main memory 110 using an index vector is referred to as a scatter.

[0045] The vector processor 180 according to one embodiment may read a plurality of pieces of data at memory addresses calculated by adding a plurality of offset values stored in an index vector to a base address to perform a gather thereon, and may store the plurality of pieces of data in the vector register 160 in the form of a vector.

[0046] For example, in the vector processor 180 that performs an 8-way vector operation, eight indexes, that is, relative addresses of a memory in which eight pieces of data are stored, may be stored in an index vector, and the vector processor 180 may read data of a corresponding address by adding the eight indexes to the base address. The eight pieces of read data are stored in the vector register 160 in the form of a vector.

[0047] A process of performing a scatter is a reverse process of a gather. The vector stored in the vector register 160 is divided into eight elements and the eight elements are stored at memory addresses obtained by adding offsets of the index vector to the base address.

[0048] However, when the vector processor 180 simultaneously reads or stores a plurality of pieces of data while performing a gather and a scatter, the vector processor 180 may frequently access the same memory bank. In this case, a memory bank conflict may cause a stall, and thus performance and efficiency of the vector processor may be significantly degraded.

[0049] As described above, the vector processor 180 needs to simultaneously access the main memory 110 to read the plurality of pieces of data from the main memory 110 and store the plurality of pieces of data in the main memory 110. Therefore, in order for the vector processor 180 to simultaneously access the main memory 110, the main memory 110 may have a multi-memory bank structure, and the vector processor 180 may additionally require a memory system (not illustrated) that may independently control addresses of memory banks.

[0050] FIG. 2 is a diagram illustrating a method by which a vector processor according to one embodiment performs a gather and a scatter.

[0051] The vector processor 180 may generate a plurality of lookup tables (shaded portions) 210, 220, 230, 240, 250, 260, 270, and 280 in which data that is necessary for a vector operation among pieces of data stored in the main memory 110 is copied and stored. The main memory 110 has a multi-bank structure including a plurality of memory banks.

[0052] The vector processor 180 may gather a plurality of pieces of data stored at a predetermined index position of each of the lookup tables from the generated lookup tables 210, 220, 230, 240, 250, 260, 270, and 280, and generate the plurality of pieces of data in the form of a vector.

[0053] The vector processor 180 according to one embodiment may gather a plurality of pieces of data A, B, C, D, E, F, G, and H from the lookup tables 210, 220, 230, 240, 250, 260, 270, and 280, respectively. The plurality of pieces of gathered data A, B, C, D, E, F, G, and H may be stored in the vector register 160 in the form of a vector, and the vector in the vector register 160 may be an operand of a vector operation.

[0054] The vector processor 180 may divide the vector stored in the vector register 160 into elements and scatter the vector divided by elements at a predetermined index position in each of the lookup tables 210, 220, 230, 240, 250, 260, 270, and 280. The vector in the vector register 160, which is an object of the scattering, may be result vector data obtained by performing a predetermined vector operation on the gathered and generated vector data.

[0055] As described above, when a memory bank conflict occurs while the vector processor 180 accesses the plurality of banks of the main memory 110 to perform a gather and a scatter, performance of the vector processor 180 is significantly degraded. Hereinafter, methods in which the vector processor 180 according to one embodiment performs a gather and a scatter using lookup tables to reduce the number of memory bank conflicts will be described in detail with reference to FIGS. 3 to 10.

[0056] FIG. 3 is a diagram illustrating a structure of a vector processor according to one embodiment.

[0057] As described above, the main memory 110 may have a multi-bank structure. For convenience of description, it is assumed that the main memory 110 includes eight banks 120, 121, 122, 123, 124, 125, 126, and 127 and a vector processor 180 performs an 8-way vector operation.

[0058] In FIG. 3, it is illustrated that the vector processor 180 performs an 8-way vector operation and the vector register 160 stores an 8-way vector, but the inventive concept is not limited thereto. It should be apparent that the vector processor 180 and the vector register 160 may process and store an n-way vector.

[0059] The vector processor 180 according to one embodiment may include a gather unit 320 and a lookup table generating unit 340. Also, the vector processor 180 may include a controller (not illustrated) including the gather unit 320 and the lookup table generating unit 340.

[0060] The lookup table generating unit 340 according to one embodiment may generate a plurality of lookup tables. The lookup table generating unit 340 may generate one lookup table for each of the banks 120, 121, 122, 123, 124, 125, 126, and 127 in the main memory 110, or may group a predetermined number of banks into one group and generate

one lookup table for each group. That is, the lookup table generating unit 340 may divide the plurality of banks in the main memory 110 into N groups and generate one lookup table for each of the groups. N is an integer greater than 1.

[0061] The lookup table refers to a table in which data that is necessary for a vector operation currently being performed in the main memory 110 is copied and stored. For example, improving quality of an 8K ultra high definition (UHD) image is required, 16 x 16 Gaussian filters may be applied in a raster order of the image. The 256 Gaussian coefficients necessary for applying the 16 x 16 Gaussian filters may be scattered and stored in the main memory 110. The lookup table generating unit 340 according to one embodiment may generate a plurality of lookup tables, in which pieces of data for all Gaussian coefficients which are scattered and present in the main memory 110 are copied and stored, in an initialization step before performing image quality improvement.

[0062] The lookup table generating unit 340 according to one embodiment may generate identical lookup tables for the eight banks 120, 121, 122, 123, 124, 125, 126, and 127 as illustrated in FIG. 4. In this case, eight lookup tables are generated, and the eight generated lookup tables may be stored in the eight banks 120, 121, 122, 123, 124, 125, 126, and 127, respectively.

[0063] Also, as illustrated in FIG. 5, the lookup table generating unit 340 according to one embodiment may group the banks of the main memory 110 into a first group (including a bank 0 120 and a bank 1 121), a second group (including a bank 2 122 and a bank 3 123), a third group (including a bank 4 124 and a bank 5 125), and a fourth group (including a bank 6 126 and a bank 7 127), and generate one lookup table for each of the groups. In this case, four lookup tables are generated, and the four generated lookup tables may be divided and stored in two banks.

[0064] Also, the lookup table generating unit 340 according to one embodiment may group the banks of the main memory 110 into a first group (including a bank 0 120, a bank 1 121, a bank 2 122, and a bank 3 123), and a second group (including a bank 4 124, a bank 5 125, a bank 6 126, and a bank 7 127), and generate one lookup table for each of the groups. In this case, two lookup tables are generated.

[0065] The lookup table generating unit 340 according to one embodiment may determine the number of lookup tables that will be generated in consideration of a space in the main memory 110, which stores the lookup table, and the number of memory bank conflicts occurring when the vector processor 180 accesses the lookup tables. That is, since each of the lookup tables requires a storage space in the main memory 110 and the number of conflicts occurring in the banks decreases as the number of the generated lookup tables increases, there is a trade-off between the number of lookup tables and the number of memory bank conflicts.

[0066] In other words, when the lookup table generating unit 340 generates lookup tables corresponding to the number of the banks 120, 121, 122, 123, 124, 125, 126, and 127 and the gather unit 320 generates a vector including the same number of elements as the number of banks, only one access is performed on one memory bank even when a plurality of pieces of data are simultaneously randomly read. In this case, there is no memory bank conflict but a space which stores the lookup tables is maximized. However, when one lookup table is generated for each predetermined number of banks, the number of memory bank conflicts may be increased but the space which stores the lookup tables may be reduced.

[0067] When the performance of the vector processor 180 is important, the lookup table generating unit 340 may generate the maximum number of lookup tables (i.e., for each bank) and may group a predetermined number of banks into one group to generate one lookup table for each group in consideration of a storage space in which the lookup table will be stored. When the lookup table generating unit 340 groups the predetermined number of banks into one group to generate one lookup table for each group, the lookup tables may be stored in the plurality of memory banks in an interleaving form. The process of the lookup table generating unit will be described in detail with reference to FIG. 5.

[0068] The gather unit 320 may read a plurality of pieces of data from the main memory 110 using the lookup tables. The plurality of pieces of data may be stored in the vector register 160 in the form of a vector. The gather unit 320 may access an index position of the each of lookup tables present in the main memory 110 using an index vector including a plurality of indexes randomly generated when accessing the plurality of lookup tables.

[0069] For example, the gather unit 320 may generate an 8-way vector using an index vector including eight indexes. More specifically, the gather unit 320 may access a predetermined index position of each of the lookup tables using a memory address obtained by adding each offset stored in the index vector to a base address to read data therefrom.

[0070] The gather unit 320 may read the data from the plurality of lookup tables and generate vector data by gathering all of the data constituting the vector after a predetermined reading cycle.

[0071] FIG. 4 is a diagram illustrating a plurality of lookup tables generated in a main memory according to one embodiment.

[0072] The main memory 110 according to one embodiment has a size of 512 KB (a range from 0x800x_0000 to 0x800x_FFFF), and each of the banks 120, 121, 122, 123, 124, 125, 126, and 127 has a size of 64 KB. Values of 0x8000x to 0x807x illustrated at the tops of the banks 120, 121, 122, 123, 124, 125, 126, and 127 in FIG. 4 are respectively referred to as memory start addresses of the banks.

[0073] As described above, the lookup table may be generated for each of the eight banks, or may be generated for each group by grouping a predetermined number of banks into one group. A plurality of lookup tables may be generated

by the lookup table generating unit 340.

[0074] FIG. 4 illustrates a case in which a lookup table is generated for each of eight banks, and FIG. 5 illustrates a case in which one lookup table is generated for two banks.

[0075] In FIG. 4, only lookup tables 410, 415, and 420 for the bank 0 120, the bank 3 123, and the bank 7 127 among the eight banks are illustrated and the remaining lookup tables are omitted. Also, for convenience of description, data in each of the lookup tables 410, 415, and 420 is illustrated as having the same value as each index.

[0076] The lookup table generating unit 340 according to one embodiment may generate lookup tables in the form of a two-dimensional (2D) array. That is, when the number of generated lookup tables is I and the number of pieces of data that is necessary for a vector operation is J, a lookup table may be generated as a 2D array of I x J. The lookup tables may be distinguished by different top addresses.

[0077] The 2D array may be generated in the initialization step before the vector processor 180 performs a vector operation. In FIG. 4, since lookup tables are generated for all eight of the banks 120, 121, 122, 123, 124, 125, 126, and 127, eight lookup tables may be generated. Since each of the lookup tables includes 256 pieces of data, a 2D array of 8 x 256 may be declared.

[0078] The vector processor 180 according to one embodiment may perform a gather using software codes illustrated in the following Table 1. It should be apparent that the software codes in Table 1 are only one embodiment for performing the gather and may be implemented in other forms.

[Table 1]

```
ushort8 _I_intr_gather8_uh(ushort8 _I src1, uchar** _I src2)
{
ushort8 ret;
     ret = (ushort8)(0, 0, 0, 0, 0, 0, 0, 0);
      ret.s0 = _I_src2[0][_I_src1.s0];
      ret.s1 = _I_src2[1][_I_src1.s1];
      ret.s2 = _I_src2[2][_I_src1.s2];
      ret.s3 = _I_src2[3][_I_src1.s3];
      ret.s4 = _I_src2[4][_I_src1.s4];
      ret.s5 = _I_src2[5][_I_src1.s5];
      ret.s6 = _I_src2[6][_I_src1.s6];
      ret.s7 = _I_src2[7][_I_src1.s7];
   return ret;
}
```

[0079] Referring to the codes in Table 1, "ret" denotes a vector generated by performing a gather operation. The "ret" vector denotes an 8-way vector including unsigned short type data. That is, an "I_intr gather8_uh" function illustrated in Table 1 may read data from eight banks using eight indexes to generate a vector.

[0080] "_I_src1" denotes an 8-way index vector including eight indexes composed of unsigned short type data. "_I_src1.s0" to "_I_src1.s7" may denote index positions at which pieces of data which will be read from the each of the lookup tables are present, and may be randomly generated indexes.

[0081] "_I_src2" denotes lookup tables declared as an unsigned character type 2D array. Only one access is performed on each of the lookup tables (e.g., "_I_src2[0 : 7]"). For example, data may be read from a lookup table of "_I src2[0]" using an index of "_I_src1.s0," and data may be read from a lookup table of "_I_src2[1]" using an index of "_I_src1.s1." That is, data may be read using different indexes for each lookup table. That is, since eight pieces of data are read from different banks even when a plurality of pieces of data are read from the main memory 110 using the randomly generated index vector, a memory bank conflict does not occur.

[0082] FIG. 5 is a diagram illustrating a plurality of lookup tables generated in a main memory according to one embodiment.

[0083] That is, a lookup table 530 for the bank 0 120 and the bank 1 121 may be generated, a lookup table 540 for the bank 2 122 and the bank 3 123 may be generated, a lookup table 550 for the bank 4 124 and the bank 5 125 may be generated, and a lookup table 560 for the bank 6 126 and the bank 7 127 may be generated.

[0084] Each of the four generated lookup tables is divided and stored in two banks. For example, the lookup table 530 may be divided and stored in the bank 0 120 and the bank 1 121, and the lookup table 550 may be divided and stored in the bank 4 124 and the bank 5 125.

[0085] The lookup table according to one embodiment may be stored in a plurality of banks in an interleaving form510. That is, since pieces of data located in adjacent memories are alternately stored in two different banks, memory bank conflicts may be minimized. For example, pieces of data of indexes of {0, 2, 4, ..., 252, and 254} may be stored in the bank 0 120 and pieces of data of indexes of {1, 3, 5, ..., 253, and 255} may be stored in the bank 1 121.

[0086] The vector processor 180 according to one embodiment may perform a gather operation using software codes illustrated in the following Table 2. It should be apparent that the software codes in Table 2 are only one embodiment for performing the gather operation and may be implemented in other forms.

[Table 2]

```
ushort8 _I_intr_gather8_uh(ushort8 _I_src1, uchar** _I_src2)
{
ushort8 ret;
    ret = (ushort8)(0, 0, 0, 0, 0, 0, 0, 0);
      ret.s0 = _I_src2[0][_I_src1.s0];
      ret.s1 = _I_src2[0][_I_src1.s1];
      ret.s2 = _I_src2[1][_I_src1.s2];
      ret.s3 = _I_src2[1][_I_src1.s3];
      ret.s4 = _I_src2[2][_I_src1.s4];
      ret.s5 = _I_src2[2][_I_src1.s5];
      ret.s6 = _I_src2[3][_I_src1.s6];
      ret.s7 = _I_src2[3][_I_src1.s7];
      return ret;
}
```

[0087] In the codes in Table 2, since variables are the same as those in Table 1, descriptions thereof will be omitted. A difference from Table 1 is that two elements among eight elements constituting the finally returned vector "ret" are generated by referring to the same lookup table. For example, first and second elements of "ret" are pieces of data which are read by referring to the lookup table 530, and third and fourth elements thereof are pieces of data which are read by referring to the lookup table 540.

[0088] Since the plurality of elements refer to one lookup table when the number of generated lookup tables is reduced, the number of memory bank conflicts is increased. However, as described above, when the one lookup table present in the plurality of banks is implemented in an interleaving form, the number of memory bank conflicts may be minimized.

[0089] As illustrated in FIG. 5, the lookup table is implemented in an interleaving form 510 in which pieces of data of even-numbered indexes are stored in the bank 0 120, the bank 2 122, the bank 4 124, and the bank 6 126 and pieces of data of odd-numbered indexes are stored in the bank 1 121, the bank 3 123, the bank 5 125, and the bank 7 127. When a calling index vector is an 8-way vector including {an even number, an odd number, an even number, an odd number, an even number, an odd number, an even number, an odd number}, a memory bank conflict does not occur.

[0090] FIG. 6A is a diagram illustrating a structure of a vector processor according to one embodiment.

[0091] The vector processor 180 according to one embodiment may include the gather unit 320, the lookup table generating unit 340, and a scatter unit 630. Since the gather unit 320 and the lookup table generating unit 340 have been described above with reference to FIG. 3, descriptions thereof will be omitted.

[0092] The scatter unit 630 performs a process corresponding to a reverse process of the process performed by the gather unit 320. Therefore, even though the content of the gather unit 320 and the lookup table generating unit 340 described in FIGS. 2 to 5 is omitted below, the content is applied to the scatter unit 630 according to the embodiment of FIG. 6 in the same manner.

[0093] The scatter unit 630 may divide vector data stored in the vector register 160 into elements and store the vector divided by elements back in the plurality of lookup tables. Result vector data obtained by performing a predetermined vector operation on the vector data generated by the gather unit 320 may be stored in the vector register 160.

[0094] The scatter unit 630 according to one embodiment may perform a scatter using software codes illustrated in the following Table 3. It should be apparent that the software codes in Table 3 are only one embodiment for performing the scatter operation and may be implemented in other forms.

[Table 3]

```
void _I_intr_scatter8_uh(ushort8 _I_src1, ushort** _I_src2, ushort8 _I_src3)
{
        _I_src2[0][_I_src1.s0] = _I_src3.s0;
        _I_src2[1][_I_src1.s1] = _I_src3.s1;
        _I_src2[2][_I_src1.s2] = _I_src3.s2;
        _I_src2[3][_I_src1.s3] = _I_src3.s3;
        _I_src2[4][_I_src1.s4] = _I_src3.s4;
        _I_src2[5][_I_src1.s5] = _I_src3.s5;
        _I_src2[6][_I_src1.s6] = _I_src3.s6;
        _I_src2[7][_I_src1.s7] = _I_src3.s7;
}
```

[0095] Referring to the codes in Table 3, a "_intr_scatter8_uh" function may divide a vector (an 8-way vector) stored in the vector register 160 into elements and store the vector divided by elements at a predetermined position of the lookup table generated for each bank. "_I_src3" denotes a vector stored in the vector register 160, and may store a result vector obtained by performing a vector operation. "_I_src3" denotes an 8-way vector including unsigned integer type data.

[0096] "_I_src1" denotes an 8-way index vector including unsigned short type data. For example, the "_intr_scatter8_uh" function may store "_I_src3.s3" at a position of "_I_src1.s3" of a lookup table "_I_src2[3]."

[0097] FIG. 6B is a diagram illustrating a method by which a vector processor according to one embodiment synchronizes a plurality of lookup tables.

[0098] As described above, since the lookup table generating unit 340 generates a plurality of the identical lookup tables in the initialization step, pieces of data of the plurality of lookup tables may be synchronized with each other. That is, unlike the gather unit 320, the scatter unit 630 may change values stored at a predetermined index position of each of the respective lookup tables, and update values stored at the same index position of different lookup tables with the changed value. The synchronization between the lookup tables may be implemented by a switching unit 745 which will be described with reference to FIG. 7.

[0099] In other words, when values of elements of a vector stored in the vector register 160 are changed and a value stored at a predetermined index of a predetermined lookup table is updated, the scatter unit 630 may also update pieces of data stored at predetermined indexes of the remaining lookup tables with the same value.

[0100] For example, when a value of a twelfth index 620 of the bank 0 120 is changed to X after a vector operation is performed, a value of a twelfth index position of each of the remaining banks 120, 121, 123, 124, 125, 126, and 127 may also be updated with the same X.

[0101] However, the scatter unit 630 does not necessarily change the values of all of the lookup tables to the same value in all cases, and may selectively perform a scatter operation using a switching unit 745 which will be described below with reference to FIGS. 8 to 10.

[0102] FIG. 7 is a diagram illustrating a structure of a vector processor according to one embodiment.

[0103] The vector processor according to the embodiment may include the gather unit 320, the scatter unit 630, the switching unit 745, and the lookup table generating unit 340.

[0104] The switching unit 745 may control access of the gather unit 320 and the scatter unit 630 to each bank to determine whether to perform a gather and a scatter on a predetermined index position of each lookup table. The switching unit 745 may include a sub-switch unit (not illustrated) corresponding to each of the lookup tables, and the sub-switch units may determine whether to allow the gather unit 320 and the scatter unit 630 to access predetermined index positions in the banks in which the lookup tables are present.

[0105] Each of the sub-switch units may have an on/off switch for each bank in which a plurality of lookup tables are present. That is, each of the sub-switch units may include the same number of switches as the number of banks, and the switches may determine whether the gather unit 320 and the scatter unit 630 may access each of the banks.

[0106] When the sub-switch unit turns off the switch for a predetermined bank, the gather unit 320 may not read data from the turned-off bank and the scatter unit 630 may also not store data in the turned-off bank. Also, when the switch for the predetermined bank is turned on, the gather unit 320 and the scatter unit 630 may respectively perform a gather and a scatter on a predetermined index position of the turned-on bank.

[0107] For example, when there are eight banks in the main memory 110, one lookup table is generated for each bank, and there are eight lookup tables, the switching unit 745 may include eight sub-switch units, and each of the sub-switch units may include the same number of switches as the number of banks. When it is necessary to synchronize a plurality of identical lookup tables, the vector processor 180 may appropriately turn on the switches in each of the sub-switch units in the switching unit 745 and perform a scatter on the plurality of identical lookup tables. The process of vector processor 180 will be described below with reference to FIG. 9.

[0108] Also, when the vector processor 180 gathers one or two pieces of data from each of the lookup tables to generate data in the form of a vector, the vector processor 180 may set only one or two switches in each of the sub-switch units in the switching unit 745 to be turned-on and perform a gather on the one or two switches. The process of vector processor 180 will be described below with reference to FIGS. 8A and 8B.

[0109] FIG. 8A is a diagram illustrating a method by which a vector processor according to one embodiment performs a gather.

[0110] For convenience of description, it is assumed that the number of pieces of data that is necessary for a vector operation is eight and that each lookup table is stored in two banks in an interleaving form. For example, a lookup table 850 is stored in the bank 0 120 and the bank 1 121 in an interleaving form. The gather unit 320 may gather two pieces of data for each lookup table to generate an 8-way vector.

[0111] The switching unit 745 may include four sub-switch units 810, 820, 830, and 840 corresponding to the lookup tables. Each of the sub-switch units 810, 820, 830, and 840 may include the same number of switches as the number of the banks 120, 121, 122, 123, 124, 125, 126, and 127. An uppermost left switch in each of the sub-switch units 810, 820, 830, and 840 may control access to the bank 0 120, and a lowermost right switch therein may control access to the bank 7 127.

[0112] The switches which are set to be turned-on among the switches in each of the sub-switch units 810, 820, 830, and 840 are shaded. Hereinafter, the uppermost left switch in the sub-switch unit is referred to as a first switch, and the switches are numbered in order from left to right and from top to bottom. That is, the lowermost right switch in each of the sub-switch units 810, 820, 830, and 840 is an eighth switch.

[0113] The gather unit 320 according to one embodiment may read pieces of data of index 0 and index 5 of the lookup table 850 which are respectively stored in the bank 0 120 and the bank 1 121 using the sub-switch unit 810 in which the first and second switches are turned on. Also, for example, the gather unit 320 may read pieces of data of index 4 and index 1 of the lookup table 870 which are respectively stored in the bank 4 124 and the bank 5 125 using the sub-switch unit 830 in which the fifth and sixth switches are turned on.

[0114] FIG. 8B is a diagram illustrating a method by which a vector processor according to one embodiment performs a gather.

[0115] FIG. 8B illustrates a case in which a lookup table is generated for each of the banks 120, 121, 122, 123, 124, 125, 126, or 127. The switching unit 745 may include eight sub-switch units 815, 825, 835, 845, 855, 865, 875, and 885 corresponding to the respective lookup tables. Each of the sub-switch units 815, 825, 835, 845, 855, 865, 875, and 885 may include eight switches equal to the number of banks.

[0116] The vector processor 180 according to one embodiment may set only one switch among the eight switches included in each of the sub-switch units 815, 825, 835, 845, 855, 865, 875, and 885 to be turned on, and read one piece of data present at a predetermined index position of each of lookup tables 818, 828, 838, 848, 858, 868, 878, and 888.

[0117] For example, the gather unit 320 may read data stored at an index 0 of the lookup table 818 stored in the bank 0 120 using the sub-switch unit 815 in which the first switch is turned on, and may read data stored at an index 7 of the lookup table 868 stored in the bank 5 125 using the sub-switch unit 865 in which the sixth switch is turned on.

[0118] FIG. 9 is a diagram illustrating a method by which a vector processor according to one embodiment performs a scatter.

[0119] For convenience of description, it is assumed that one lookup table stores four pieces of data and is stored in two banks in an interleaving form. For example, a lookup table 950 is stored in the bank 0 120 and the bank 1 121 in an interleaving form. That is, indexes 0 and 2 of the lookup tables are present in the even-number banks 120, 122, 124, and 126, and indexes 1 and 3 thereof are present in the odd-number banks 121, 123, 125, and 127.

[0120] The scatter unit 630 according to one embodiment may divide a vector stored in the vector register 160 into elements and scatter and store the vector divided by elements in lookup tables 950, 960, 970, and 980.

[0121] For example, it is assumed that a vector previously stored in the vector register 160 is a 4-way vector and all elements of the vector are updated after a vector operation is performed. The scatter unit 630 may store a first element of the 4-way vector in the lookup table 950, store a second element in the lookup table 960, store a third element in the lookup table 970, and store a fourth element in the lookup table 980. That is, the four elements may be scattered and stored in different indexes of the respective lookup tables 950, 960, 970, and 980.

[0122] However, as described above, synchronization of a plurality of lookup tables may be required. The vector processor 180 may appropriately set eight switches present in each of sub-switch units 910, 920, 930, and 940 to be turned on, and perform a scatter on all index positions in the respective lookup tables.

[0123] That is, the first, third, fifth, and seventh switches of the sub-switch unit 910 may be turned on to store data in the index 2 of each of the lookup tables 950, 960, 970, and 980.

[0124] Also, the second, fourth, sixth, and eighth switches of the sub-switch unit 920 may be turned on to store data in the index 3 of each of the lookup tables 950, 960, 970, and 980.

[0125] Also, the first, third, fifth, and seventh switches of the sub-switch unit 930 may be turned on to store data in the index 0 of each of the lookup tables 950, 960, 970, and 980.

[0126] Also, the second, fourth, sixth, and eighth switches of the sub-switch unit 940 may be turned on to store data in the index 1 of each of the lookup tables 950, 960, 970, and 980.

[0127] As a result, all pieces of data stored in the indexes 0 to 3 in each of the lookup tables 950, 960, 970, and 980 may be updated.

[0128] It should be apparent that the vector processor 180 may set only some of the switches of each of the sub-switch units 910, 920, 930, and 940 to be turned on and perform a scatter on the look up tables when there is no need to synchronize the lookup tables. For example, although not illustrated in FIG. 9, only the first switch of the sub-switch unit 910 may be turned on to store the data at a position of the index 2 of the lookup table 950 stored in the bank 0 120. Also, only the fourth switch of the sub-switch unit 920 may be turned on to store the data at a position of the index 3 of the lookup table 960 stored in the bank 3 123. Also, only the fifth switch of the sub-switch unit 930 may be turned on to store the data at a position of the index 0 of the lookup table 970 stored in the bank 4 124. Also, only the eighth switch of the sub-switch unit 940 may be turned on to store the data at a position of the index 1 of the lookup table 980 stored in the bank 7 127.

[0129] The gather unit 320, the scatter unit 630, the switching unit 745, and the lookup table generating unit 340 according to one embodiment may be implemented by generating lookup tables and expanding only a load and store operation of the vector processor without changing a structure of the vector processor 180. Therefore, the gather unit 320, the scatter unit 630, the switching unit 745, and the lookup table generating unit 340 according to the embodiment may be implemented in an intrinsic form in which an instruction set architecture (ISA) of the vector processor 180 is not changed, and thus may be implemented without modifying the structure of the vector processor 180 or without additional hardware.

[0130] Hereinafter, a method by which the vector processor 180 according to one embodiment accesses a memory and performs data gather and scatter will be described with reference to flowcharts in FIGS. 10 to 13. FIGS. 10 to 13 are flowcharts for describing a method by which the vector processor 180 illustrated in FIGS. 1 to 9 performs a gather and a scatter. Therefore, even though the content described above with respect to the vector processor 180 in FIGS. 1 to 9 is omitted, the content is applied to the gather and scatter methods according to embodiments of FIGS. 10 to 13.

[0131] FIG. 10 is a flowchart illustrating a method by which a vector processor according to one embodiment performs a gather.

[0132] In step 1020, the vector processor 180 may generate a plurality of lookup tables for a main memory. The lookup table refers to a table in which data that is necessary for a vector operation currently being performed among data stored in the main memory is copied and stored. The main memory may have a multi-bank structure. The generated lookup tables may be stored in each bank or may be divided and stored in a plurality of banks.

[0133] In step 1030, the vector processor 180 may perform a gather for reading a plurality of pieces of data from the lookup table and generating vector data. More specifically, the vector processor 180 according to one embodiment may perform a gather operation of reading a plurality of pieces of data from the main memory using the generated lookup tables and storing the plurality of pieces of data in a vector register in the form of a vector.

[0134] In step 1030, the vector processor 180 may access an index position of the each of the lookup tables present in the main memory using an index vector including a plurality of randomly generated indexes to read the data.

[0135] FIG. 11 is a flowchart illustrating a method by which a vector processor according to one embodiment generates a plurality of lookup tables.

[0136] In step 1110, the vector processor 180 may determine the number of lookup tables which will be generated. The vector processor 180 according to one embodiment may determine the number of lookup tables which will be generated in consideration of a trade-off between a space in a main memory which stores the lookup tables and the number of memory conflicts caused by the vector processor accessing the lookup table.

[0137] In step 1120, the vector processor 180 determines whether to generate the same number of lookup tables as the number of memory banks. When it is determined that the number of generated lookup tables will be equal to the number of memory banks, the process proceeds to step 1140, otherwise the process proceeds to step 1130.

[0138] In step 1130, the vector processor 180 may divide and store each of the generated lookup tables into a plurality of banks. The vector processor 180 according to one embodiment may divide and store each of the lookup tables into a plurality of banks in an interleaving form. That is, a plurality of pieces of data may be alternately stored in different banks so that memory bank conflicts may be minimized.

[0139] In step 1140, the vector processor 180 may generate one lookup table for each bank. The lookup tables are respectively stored in the banks.

[0140] FIG. 12 is a flowchart illustrating a method by which a vector processor according to one embodiment performs a scatter.

[0141] Since steps 1020 and 1030 have been described with reference to FIG. 10, descriptions thereof will be omitted.

[0142] The vector processor 180 according to one embodiment may perform a scatter for scattering and storing vector data in a lookup table.

[0143] In step 1230, the vector processor 180 according to one embodiment may divide a vector stored in a vector

register into elements to store the elements back in a lookup table. The vector stored in the vector register may perform a vector operation on the vector generated in step 1030, result vector data obtained by performing the operation is stored, and the result vector data may be stored back in a plurality of lookup tables in step 1230.

[0144] In step 1230, when the vector processor 180 according to the embodiment updates a value stored in a predetermined index of a predetermined lookup table in order to synchronize lookup tables, data stored in the predetermined index of each of the remaining lookup tables may also be updated with the same value.

[0145] FIG. 13 is a flowchart illustrating a method by which a vector processor according to one embodiment performs a gather and a scatter.

[0146] Since steps 1020, 1030, and 1230 have been described with reference to FIGS. 10 and 12, descriptions thereof will be omitted.

[0147] The vector processor 180 according to one embodiment may determine whether to allow access a predetermined index position of each lookup table in step 1310 before a gather is performed (step 1030) or a scatter is performed (step 1230). That is, the vector processor 180 may determine an index position of a lookup table on which the gather (step 1030) is to be performed or the scatter (step 1230) is to be performed by determining whether each bank stored in the lookup table is accessible.

[0148] In step 1030, vector data may be generated by reading data present at a predetermined index position that is allowed to be accessed in each lookup table. The vector processor 180 according to one embodiment may generate vector data by reading data stored at a predetermined position of a lookup table present in an accessible bank.

[0149] In step 1230, the vector data may be stored at a predetermined index position that is allowed to be accessed in each lookup table. The vector processor 180 according to one embodiment may store data at a predetermined position of a lookup table present in an accessible bank.

[0150] The vector processor 180 according to one embodiment may be applied to most processors that perform vector operations, such as a general purpose processor, a digital signal processing processor (DSP), an application specific instruction set processor (ASIP), a graphic processing unit, and the like.

[0151] Meanwhile, the method may be implemented as computer-readable codes on a computer readable recording medium. The computer readable recording medium includes all kinds of recording apparatuses in which data that may be read by a computer system is stored. For example, the computer readable recording medium includes a read only memory (ROM), a RAM, a compact disc (CD)-ROM, a magnetic tape, a floppy disk, an optical data storage, and the like, and includes a device implemented in the form of a carrier wave such as transmission over the Internet.

[0152] Also, the computer readable recording medium may be distributed to computer systems connected via network, and thus codes that may be read by processors may be stored and executed in a distributed manner.

[0153] While the inventive concept has been described in best mode embodiments thereof, alternatives, modifications and variations of the inventive concept should be apparent to those skilled in the art in view of the above description. That is, the claims should be interpreted as including all such alternatives, modifications, and variations of the inventive concept. Therefore, all content contained in the description and drawings should be interpreted as being illustrative and in a non-limiting sense.

Claims

1. An apparatus comprising:

a plurality of memory banks; and
a controller configured to generate a plurality of lookup tables in which data that is necessary for a vector operation among data stored in the plurality of memory banks is copied and stored, and to generate vector data by reading the data from the plurality of lookup tables.

2. The apparatus of claim 1, wherein the controller includes:

a lookup table generating unit configured to generate the plurality of lookup tables; and
a gather unit configured to read the data from the plurality of lookup tables and generate the vector data.

3. The apparatus of claim 1, wherein the controller generates one lookup table for each of the plurality of memory banks, wherein a number of the generated lookup table is same as a number of the plurality of memory banks.

4. The apparatus of claim 1, wherein:

the controller divides the plurality of memory banks into a predetermined number of groups and generates one

lookup table for each of the groups; and
the one lookup table generated for each of the groups is stored in a plurality of memory banks included in the group in an interleaving form.

5. The apparatus of claim 1, wherein the controller accesses each of the lookup tables using an index vector including a plurality of randomly generated indexes and reads data stored at positions of the indexes in each of the lookup tables.

6. The apparatus of claim 1, wherein the controller divides result vector data obtained by performing a predetermined vector operation on the vector data into elements and stores the result vector divided by elements in the plurality of lookup tables.

7. The apparatus of claim 6, wherein the controller stores each of the elements at a predetermined index position in each of the lookup tables.

8. The apparatus of claim 6, wherein, when a value of data stored in a first index in one of the plurality of lookup tables is changed, the controller updates data stored in a first index in each of the remaining lookup tables, whose value is not changed, with the changed value.

9. The apparatus of claim 6, further comprising a switching unit including a plurality of sub-switch units corresponding to the lookup tables,
wherein the plurality of sub-switch units determine whether to allow the controller to access a predetermined index position of each of the lookup tables.

10. The apparatus of claim 9, wherein:

the plurality of sub-switch units include a plurality of switches corresponding to each of the plurality of memory banks; and
each of the plurality of switches determines whether the controller is accessible the each of the plurality of memory banks.

11. A method comprising:

generating a plurality of lookup tables in which data that is necessary for a vector operation among data stored in a plurality of memory banks is copied and stored; and
generating vector data by reading the data from the lookup tables.

12. The method of claim 11, wherein the generating of the lookup tables includes generating one lookup table for each of the plurality of memory banks, wherein a number of the generated lookup table is same as a number of the plurality of memory banks in each memory bank.

13. The method of claim 11, wherein the generating of the lookup tables includes:

dividing the plurality of memory banks into a predetermined number of groups and generating one lookup table for each of the groups; and
storing the one lookup table generated for each of the groups in a plurality of memory banks included in the group in an interleaving form.

14. The method of claim 11, wherein the generating of the vector data by reading the data from the lookup tables includes accessing each of the lookup tables using an index vector including a plurality of randomly generated indexes and reading data stored at positions of the indexes in each of the lookup tables.

15. A computer readable recording medium in which a program for causing a computer to execute the method according to any one of claims 11 to 14 is recorded.

FIG. 1

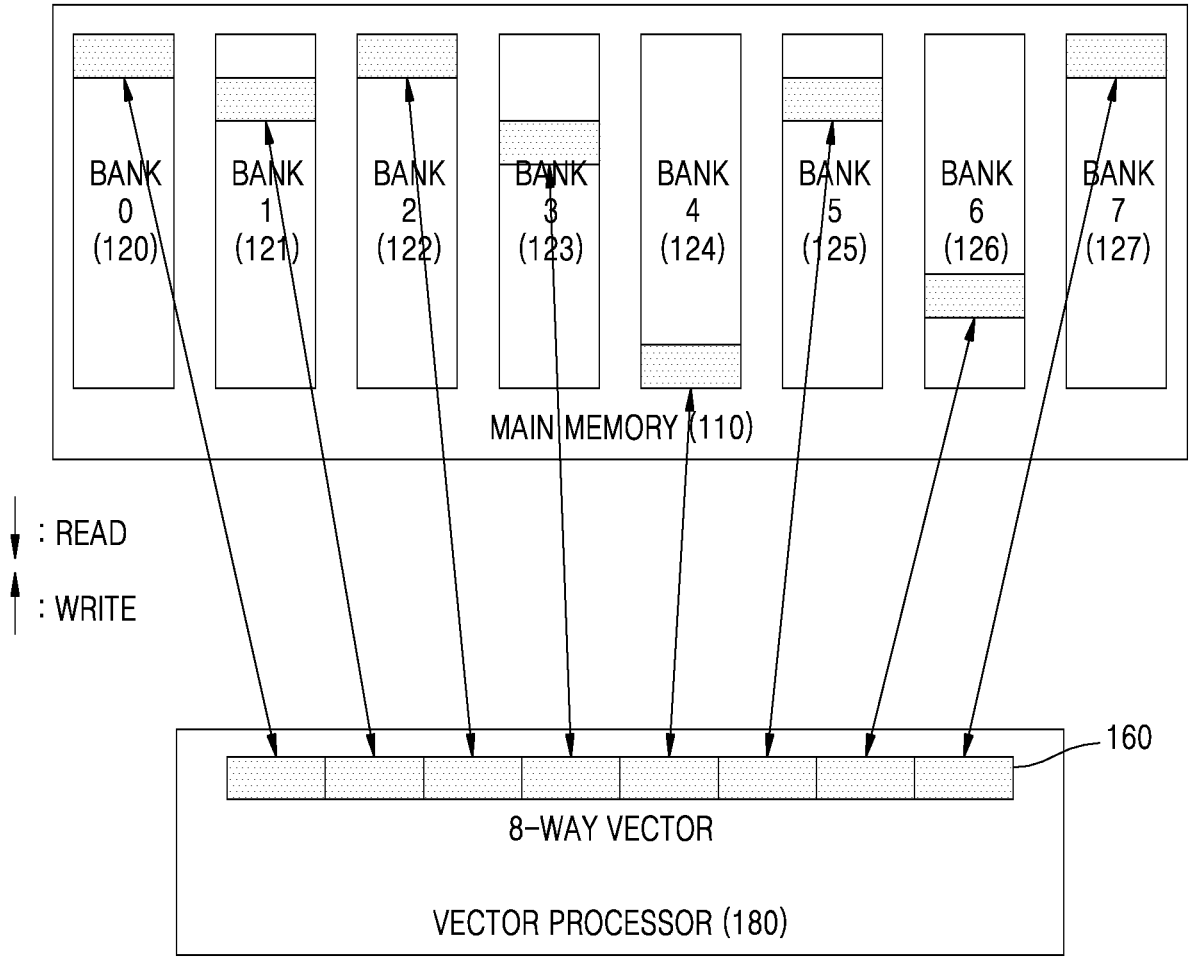

BANK 0 (120)
BANK 1 (121)
BANK 2 (122)
BANK 3 (123)
BANK 4 (124)
BANK 5 (125)
BANK 6 (126)
BANK 7 (127)
MAIN MEMORY (110)
: READ
: WRITE
160
8-WAY VECTOR
VECTOR PROCESSOR (180)

FIG. 2

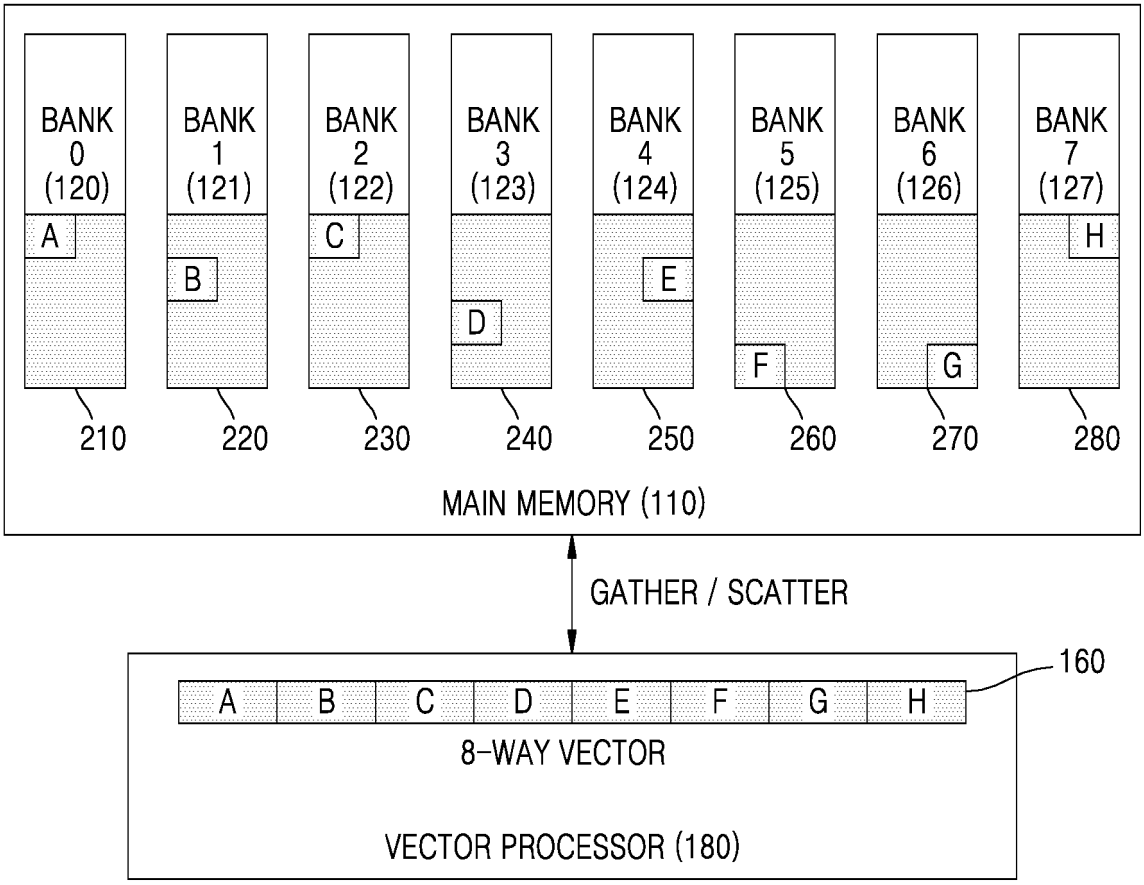

BANK 0 (120)
BANK 1 (121)
BANK 2 (122)
BANK 3 (123)
BANK 4 (124)
BANK 5 (125)
BANK 6 (126)
BANK 7 (127)
A
B
C
D
E
F
G
H
210
220
230
240
250
260
270
280
MAIN MEMORY (110)
GATHER / SCATTER
160
A
B
C
D
E
F
G
H
8-WAY VECTOR
VECTOR PROCESSOR (180)

FIG. 3

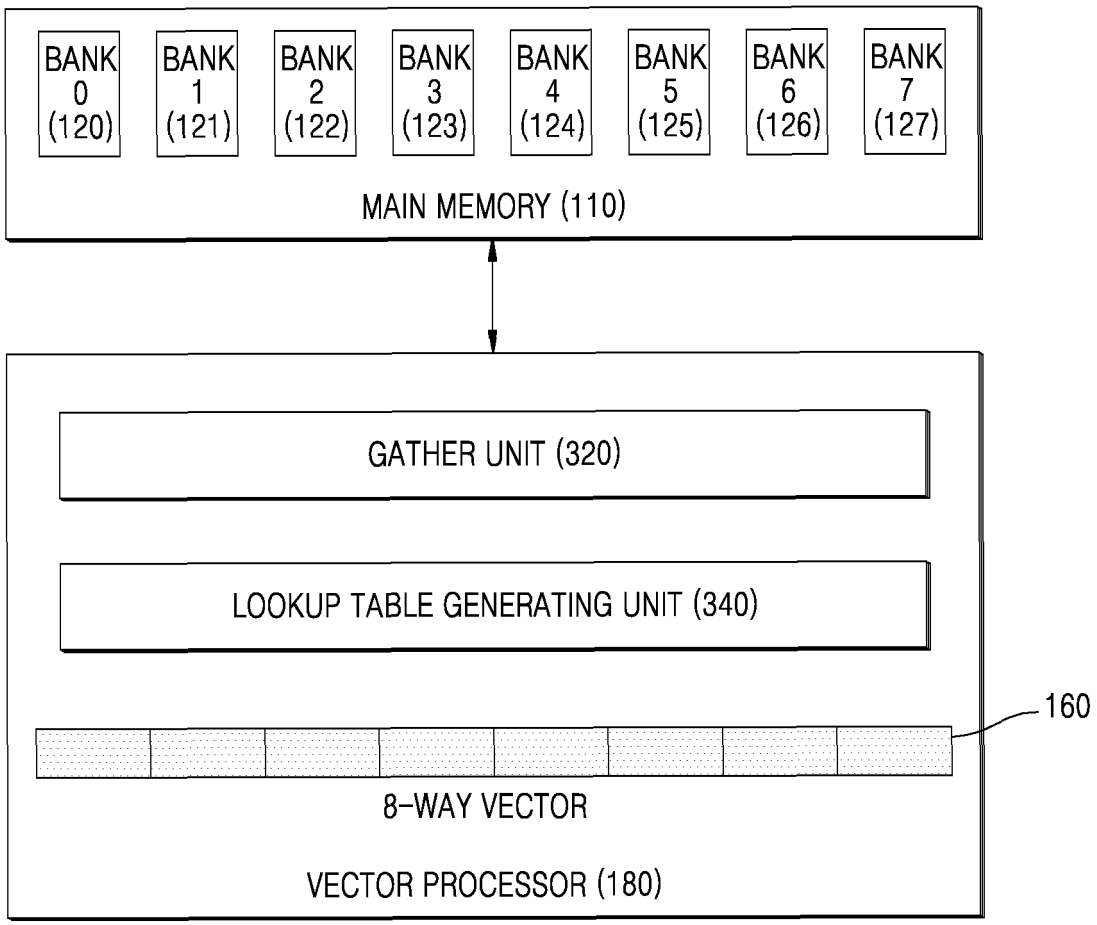

BANK 0 (120)
BANK 1 (121)
BANK 2 (122)
BANK 3 (123)
BANK 4 (124)
BANK 5 (125)
BANK 6 (126)
BANK 7 (127)
MAIN MEMORY (110)
GATHER UNIT (320)
LOOKUP TABLE GENERATING UNIT (340)
160
8-WAY VECTOR
VECTOR PROCESSOR (180)

FIG. 4

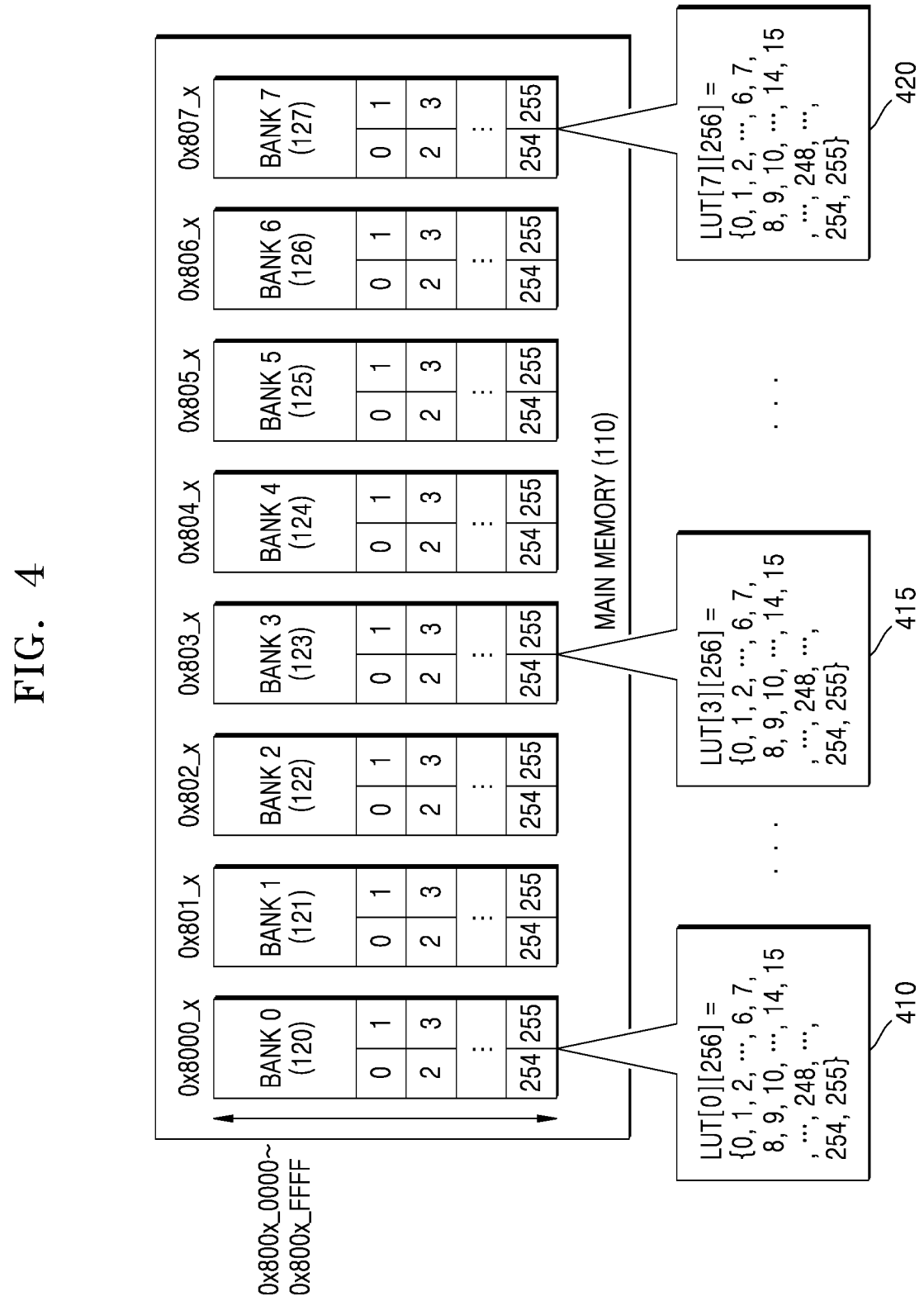

0x8000_x
0x801_x
0x802_x
0x803_x
0x804_x
0x805_x
0x806_x
0x807_x
BANK 0 (120)
BANK 1 (121)
BANK 2 (122)
BANK 3 (123)
BANK 4 (124)
BANK 5 (125)
BANK 6 (126)
BANK 7 (127)
0 1
2 3
254 255
MAIN MEMORY (110)
0x800x_0000~ 0x800x_FFFF
LUT[0][256] = {0, 1, 2, ···, 6, 7, 8, 9, 10, ···, 14, 15 , ···, 248, ···, 254, 255}
LUT[3][256] = {0, 1, 2, ···, 6, 7, 8, 9, 10, ···, 14, 15 , ···, 248, ···, 254, 255}
LUT[7][256] = {0, 1, 2, ···, 6, 7, 8, 9, 10, ···, 14, 15 , ···, 248, ···, 254, 255}
410
415
420

FIG. 5

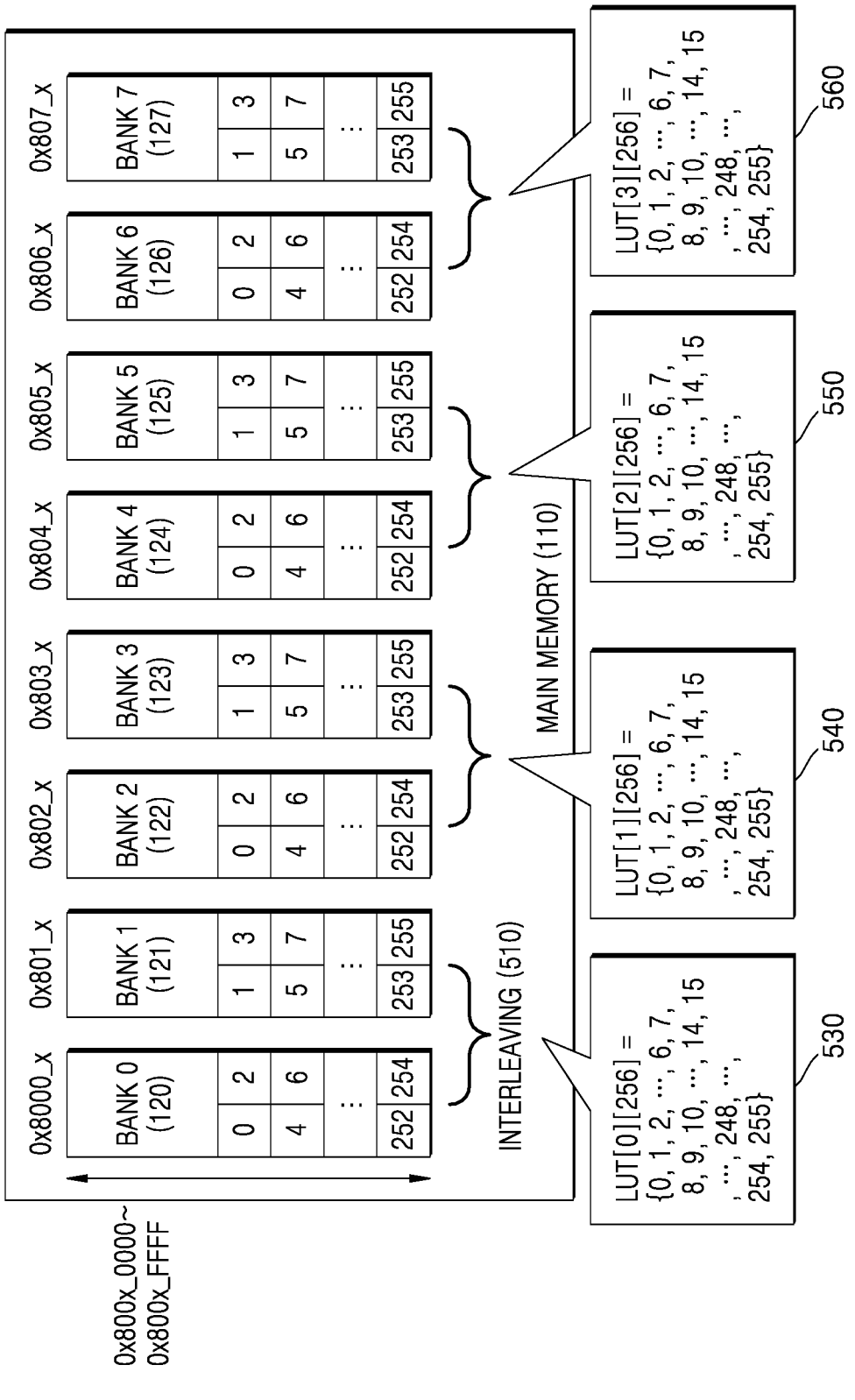

0x8000_x
0x801_x
0x802_x
0x803_x
0x804_x
0x805_x
0x806_x
0x807_x
BANK 0 (120)
BANK 1 (121)
BANK 2 (122)
BANK 3 (123)
BANK 4 (124)
BANK 5 (125)
BANK 6 (126)
BANK 7 (127)
0 2 4 6 ⋮ 252 254
1 3 5 7 ⋮ 253 255
INTERLEAVING (510)
MAIN MEMORY (110)
0x800x_0000~
0x800x_FFFF
LUT[0][256] = {0, 1, 2, ..., 6, 7, 8, 9, 10, ..., 14, 15 , ..., 248, ..., 254, 255}
LUT[1][256] = {0, 1, 2, ..., 6, 7, 8, 9, 10, ..., 14, 15 , ..., 248, ..., 254, 255}
LUT[2][256] = {0, 1, 2, ..., 6, 7, 8, 9, 10, ..., 14, 15 , ..., 248, ..., 254, 255}
LUT[3][256] = {0, 1, 2, ..., 6, 7, 8, 9, 10, ..., 14, 15 , ..., 248, ..., 254, 255}
530
540
550
560

FIG. 6A

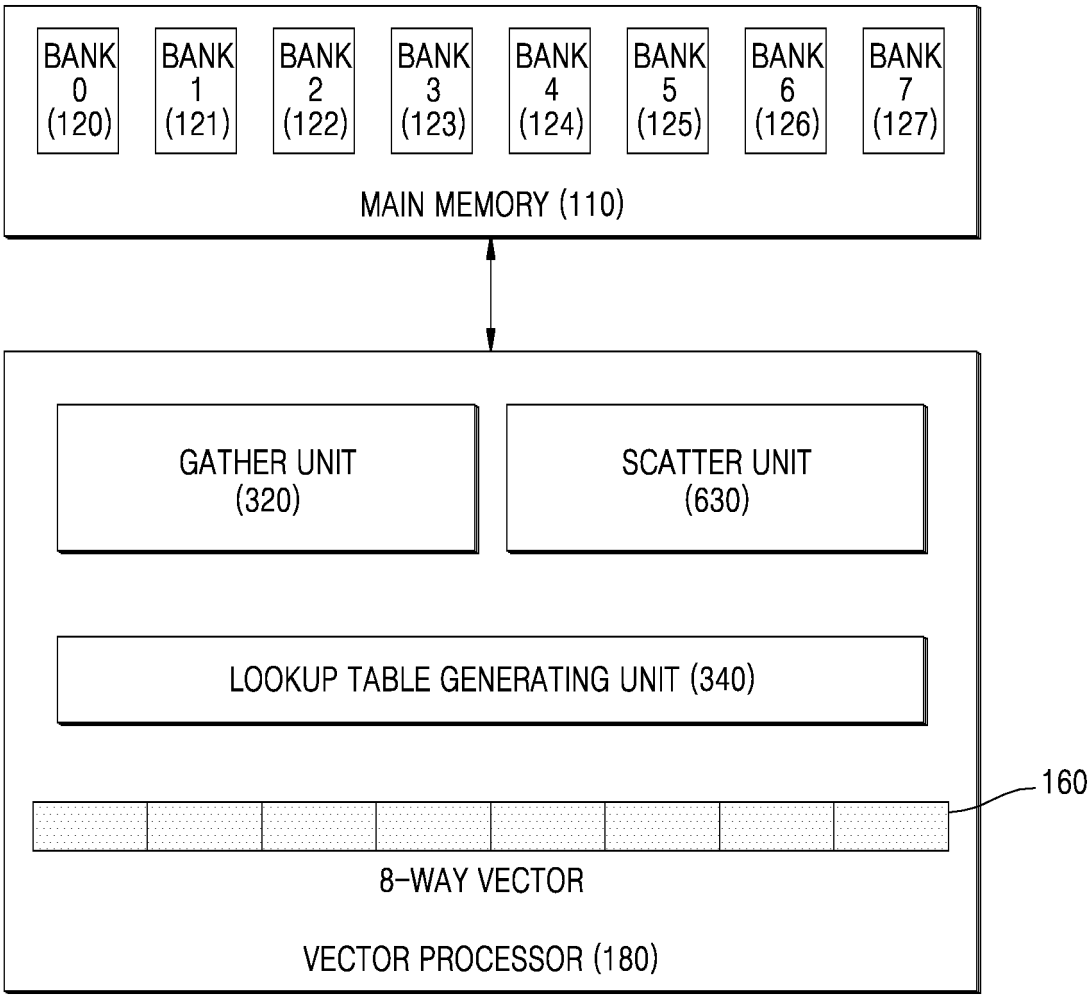

BANK 0 (120)
BANK 1 (121)
BANK 2 (122)
BANK 3 (123)
BANK 4 (124)
BANK 5 (125)
BANK 6 (126)
BANK 7 (127)
MAIN MEMORY (110)
GATHER UNIT (320)
SCATTER UNIT (630)
LOOKUP TABLE GENERATING UNIT (340)
160
8-WAY VECTOR
VECTOR PROCESSOR (180)

FIG. 6B

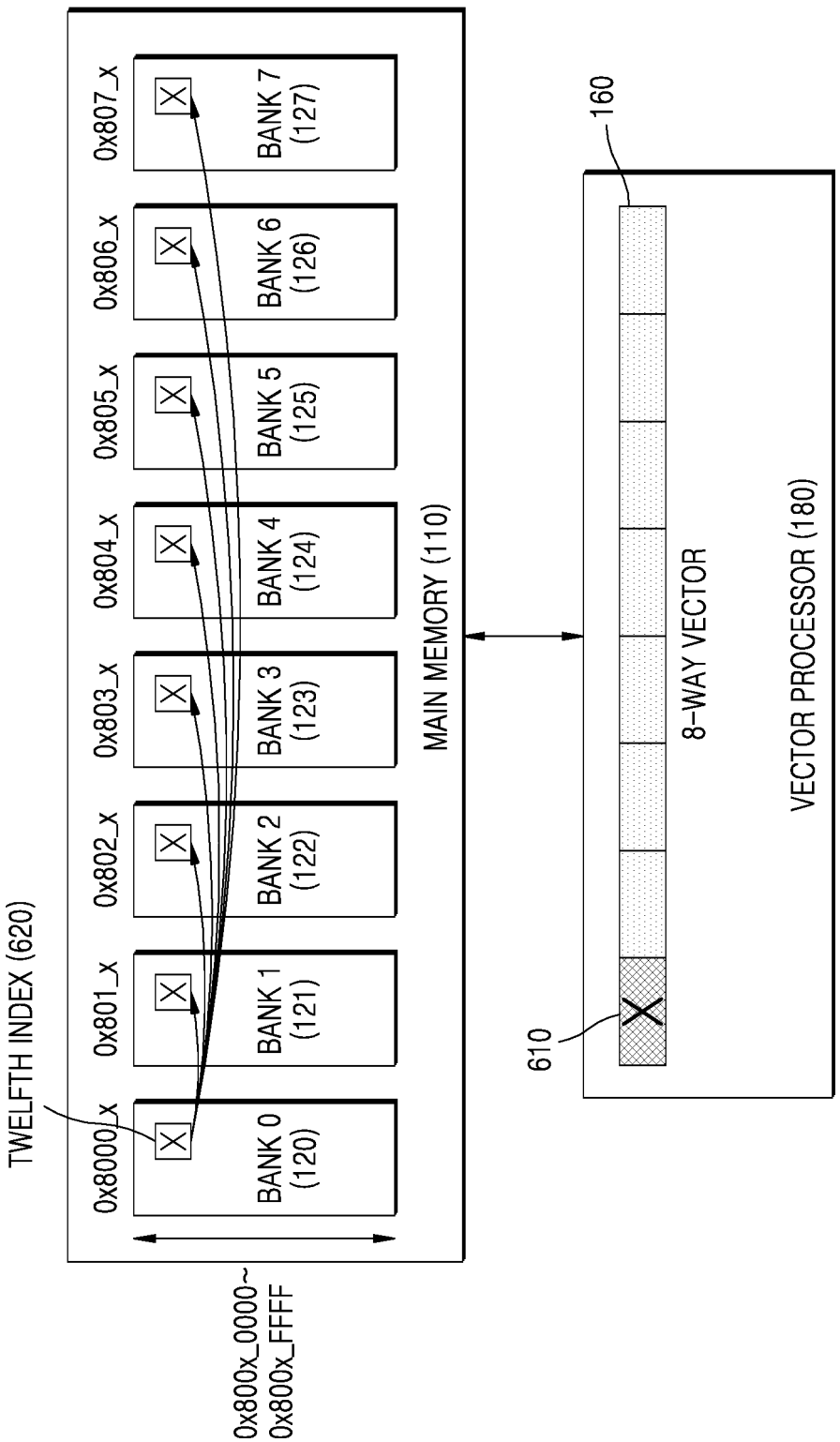

TWELFTH INDEX (620)
0x8000_x
0x801_x
0x802_x
0x803_x
0x804_x
0x805_x
0x806_x
0x807_x
BANK 0 (120)
BANK 1 (121)
BANK 2 (122)
BANK 3 (123)
BANK 4 (124)
BANK 5 (125)
BANK 6 (126)
BANK 7 (127)
MAIN MEMORY (110)
0x800x_0000~
0x800x_FFFF
160
610
8-WAY VECTOR
VECTOR PROCESSOR (180)

FIG. 7

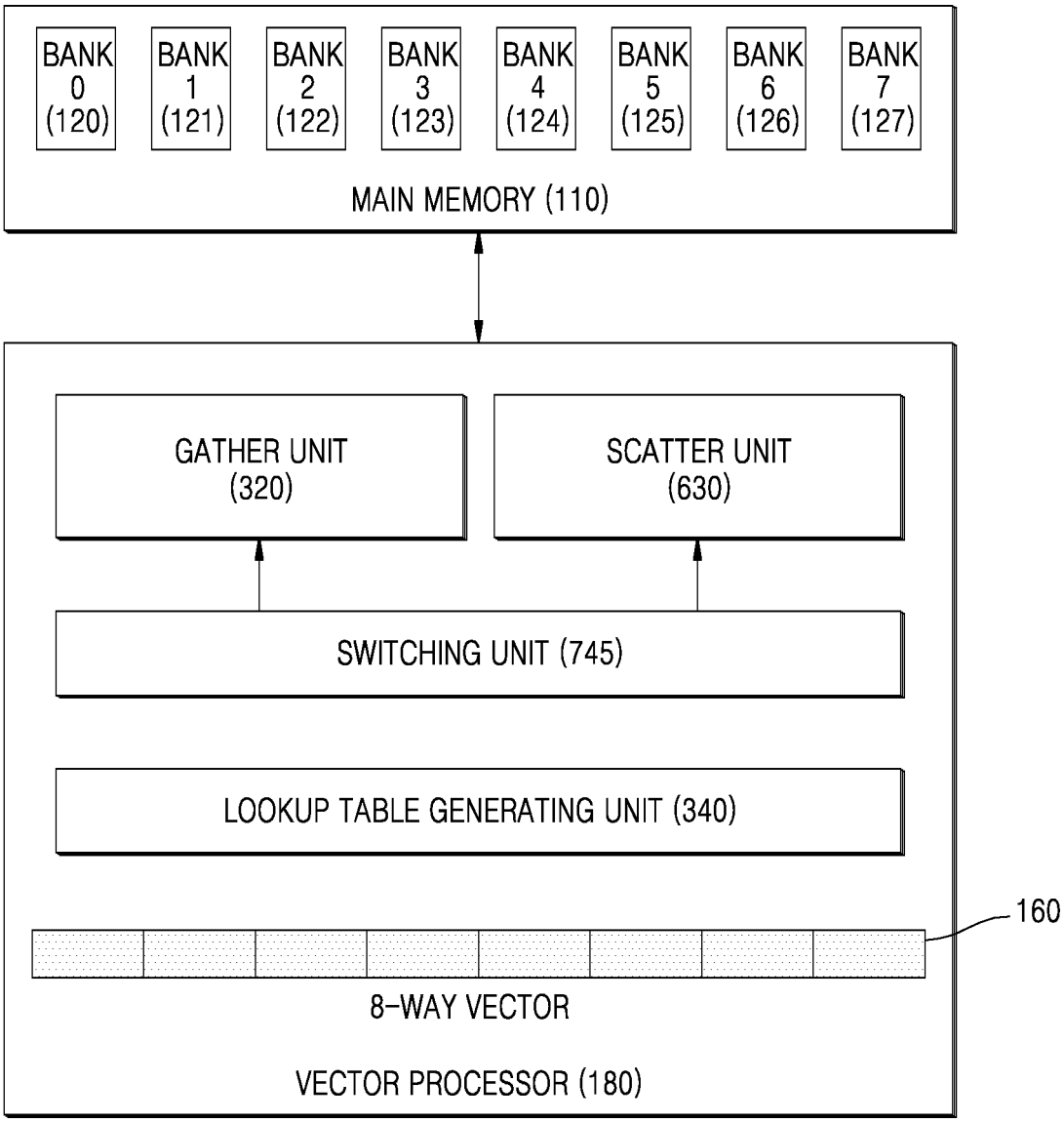

BANK 0 (120)
BANK 1 (121)
BANK 2 (122)
BANK 3 (123)
BANK 4 (124)
BANK 5 (125)
BANK 6 (126)
BANK 7 (127)
MAIN MEMORY (110)
GATHER UNIT (320)
SCATTER UNIT (630)
SWITCHING UNIT (745)
LOOKUP TABLE GENERATING UNIT (340)
160
8-WAY VECTOR
VECTOR PROCESSOR (180)

FIG. 8A

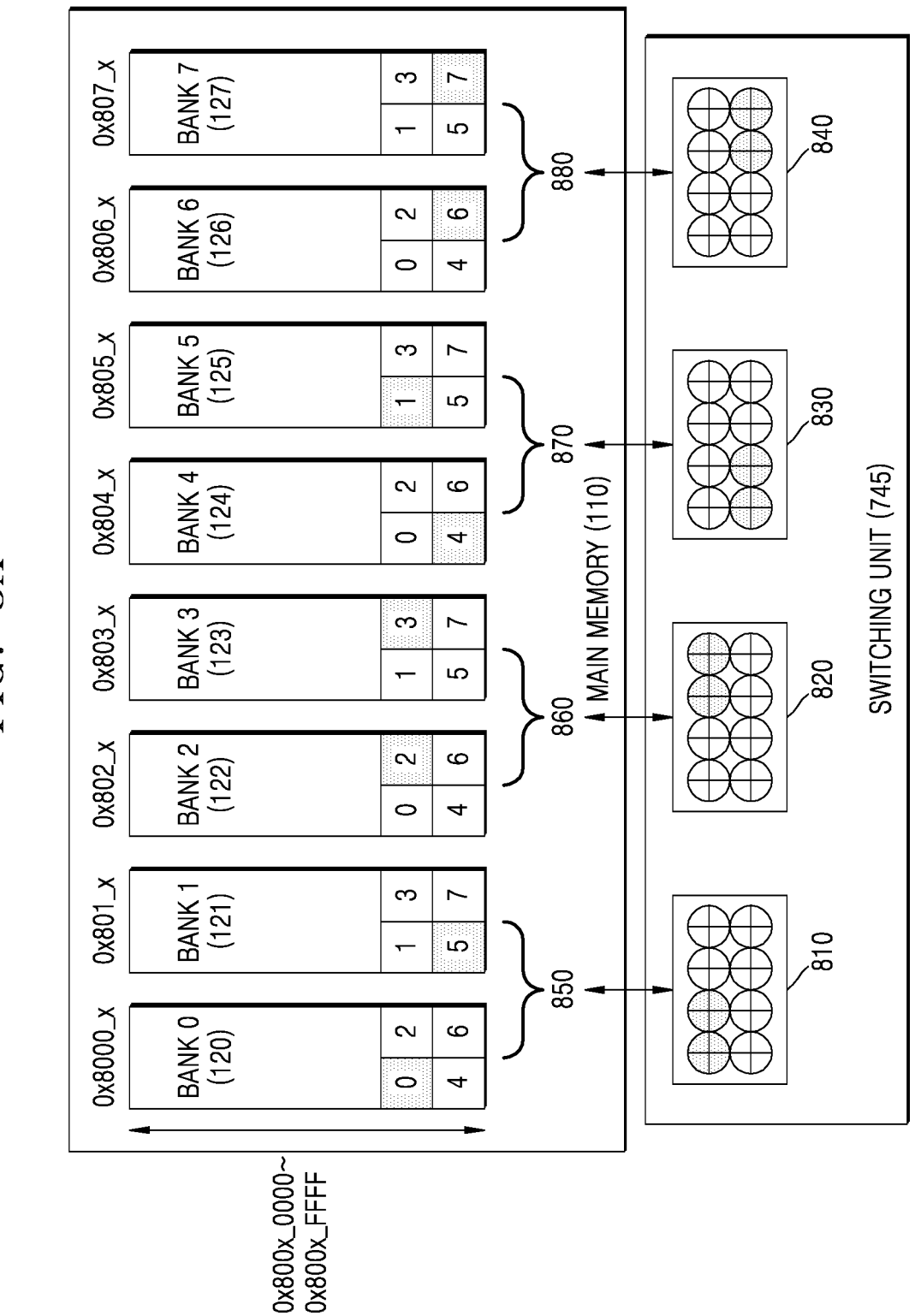

0x8000_x
0x801_x
0x802_x
0x803_x
0x804_x
0x805_x
0x806_x
0x807_x
BANK 0 (120)
BANK 1 (121)
BANK 2 (122)
BANK 3 (123)
BANK 4 (124)
BANK 5 (125)
BANK 6 (126)
BANK 7 (127)
0 2 4 6
1 3 5 7
850
860
870
880
MAIN MEMORY (110)
0x800x_0000~
0x800x_FFFF
SWITCHING UNIT (745)
810
820
830
840

FIG. 8B

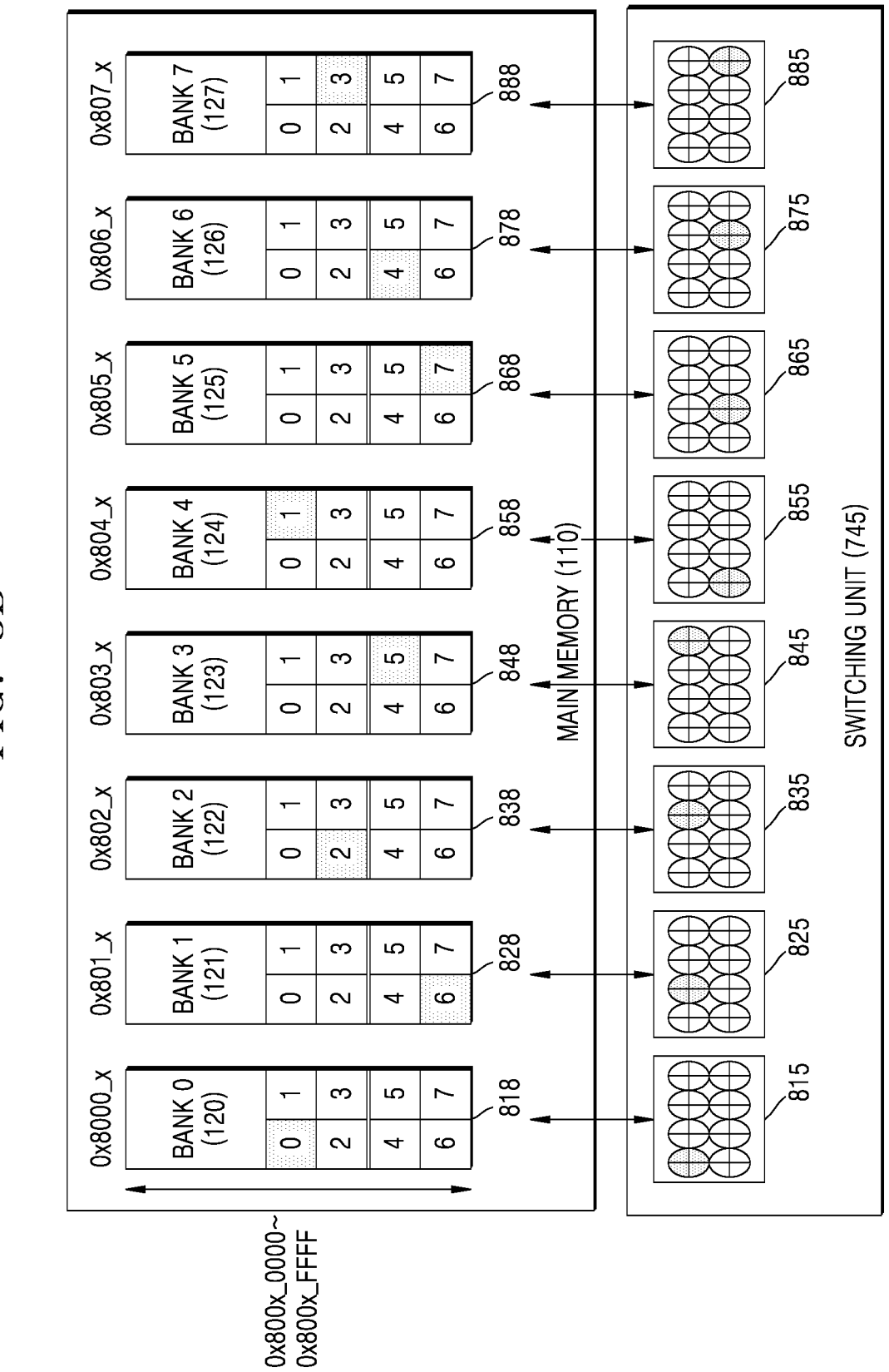

0x8000_x
0x801_x
0x802_x
0x803_x
0x804_x
0x805_x
0x806_x
0x807_x
BANK 0 (120)
BANK 1 (121)
BANK 2 (122)
BANK 3 (123)
BANK 4 (124)
BANK 5 (125)
BANK 6 (126)
BANK 7 (127)
818
828
838
848
858
868
878
888
MAIN MEMORY (110)
SWITCHING UNIT (745)
815
825
835
845
855
865
875
885
0x800x_0000~
0x800x_FFFF

FIG. 9

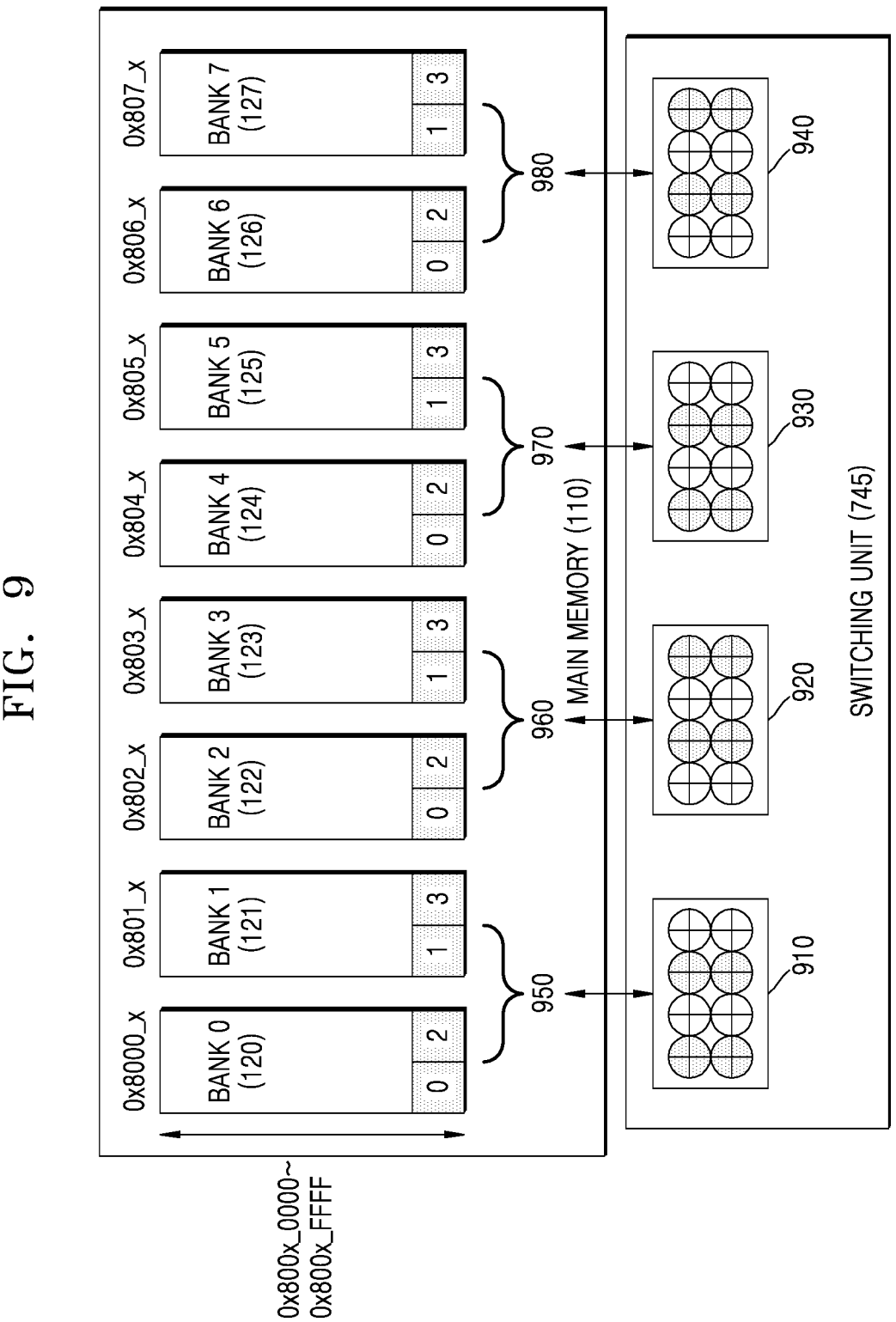

0x8000_x
0x801_x
0x802_x
0x803_x
0x804_x
0x805_x
0x806_x
0x807_x
BANK 0 (120)
BANK 1 (121)
BANK 2 (122)
BANK 3 (123)
BANK 4 (124)
BANK 5 (125)
BANK 6 (126)
BANK 7 (127)
0
1
2
3
950
960
970
980
MAIN MEMORY (110)
0x800x_0000~
0x800x_FFFF
910
920
930
940
SWITCHING UNIT (745)

FIG. 10

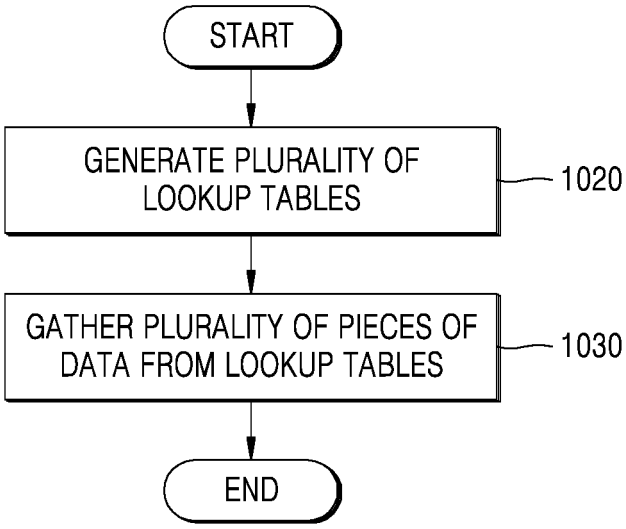

START
GENERATE PLURALITY OF LOOKUP TABLES
1020
GATHER PLURALITY OF PIECES OF DATA FROM LOOKUP TABLES
1030
END

FIG. 11

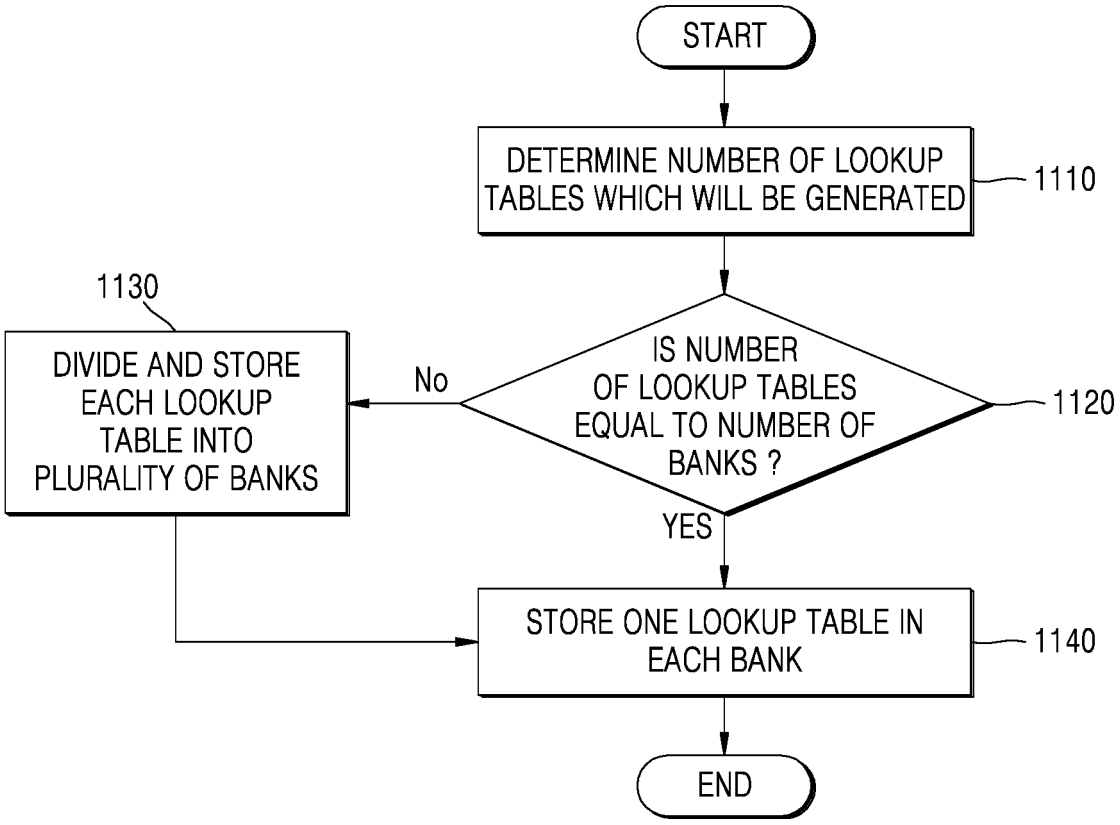

START
DETERMINE NUMBER OF LOOKUP TABLES WHICH WILL BE GENERATED
1110
IS NUMBER OF LOOKUP TABLES EQUAL TO NUMBER OF BANKS ?
1120
No
1130
DIVIDE AND STORE EACH LOOKUP TABLE INTO PLURALITY OF BANKS
YES
STORE ONE LOOKUP TABLE IN EACH BANK
1140
END

FIG. 12

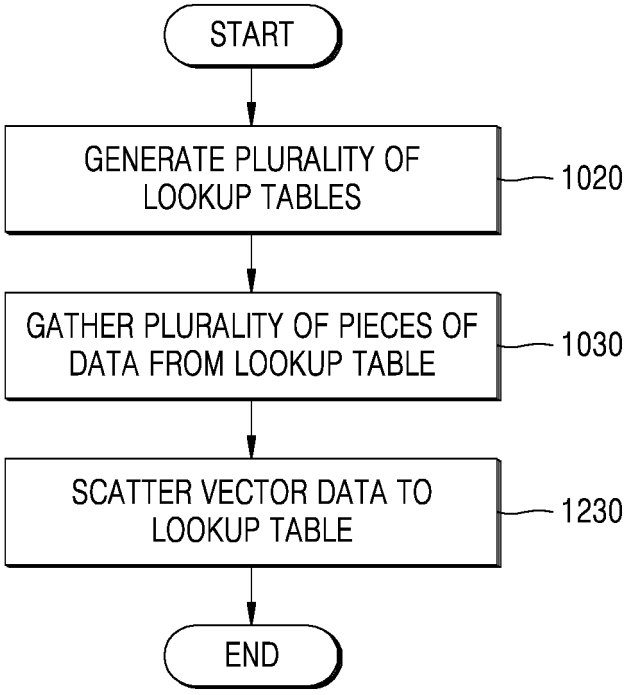
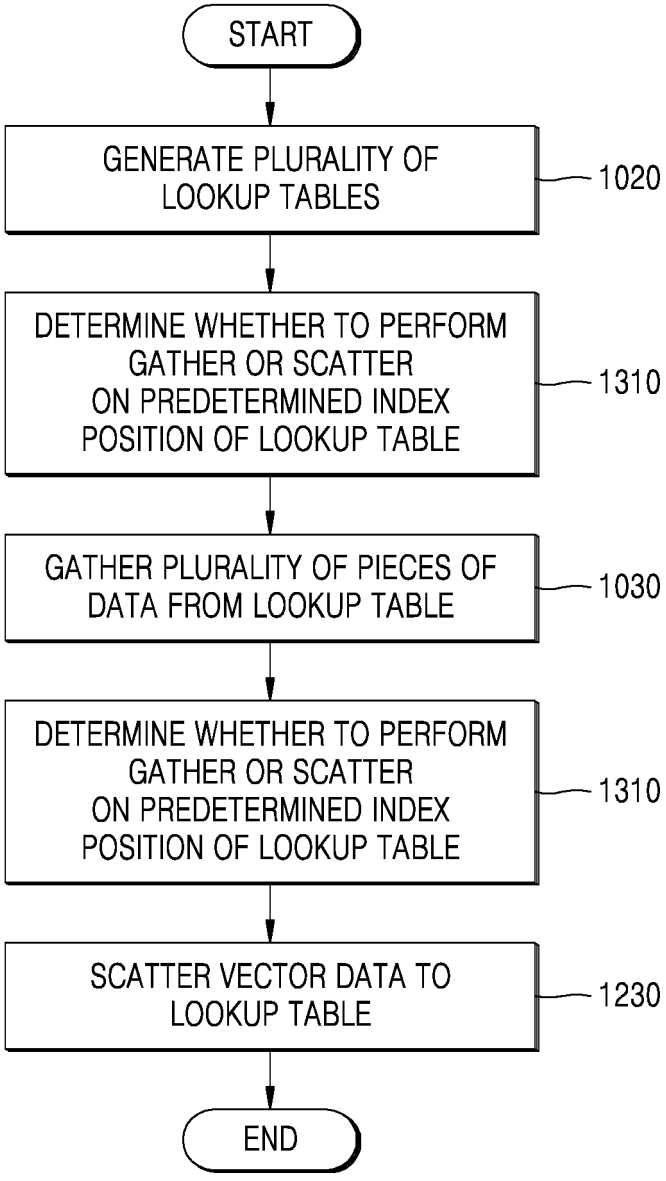

START
GENERATE PLURALITY OF LOOKUP TABLES
1020
GATHER PLURALITY OF PIECES OF DATA FROM LOOKUP TABLE
1030
SCATTER VECTOR DATA TO LOOKUP TABLE
1230
END

FIG. 13


START
GENERATE PLURALITY OF LOOKUP TABLES
1020
DETERMINE WHETHER TO PERFORM GATHER OR SCATTER ON PREDETERMINED INDEX POSITION OF LOOKUP TABLE
1310
GATHER PLURALITY OF PIECES OF DATA FROM LOOKUP TABLE
1030
DETERMINE WHETHER TO PERFORM GATHER OR SCATTER ON PREDETERMINED INDEX POSITION OF LOOKUP TABLE
1310
SCATTER VECTOR DATA TO LOOKUP TABLE
1230
END

INTERNATIONAL SEARCH REPORT

International application No.
PCT/KR2015/012317

A. CLASSIFICATION OF SUBJECT MATTER

G06F 12/02(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F 12/02; G06F 12/06; G06F 9/26; G06F 12/10; G06F 9/34; G06F 12/00

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Korean Utility models and applications for Utility models: IPC as above
Japanese Utility models and applications for Utility models: IPC as above

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
eKOMPASS (KIPO internal) & Keywords: memory, bank, lookup table, gather, copy, interleaving, group, conflict, contention

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2006-0012603 A1 (LINDHOLM, John Erik et al.) 19 January 2006 See paragraphs [0035], [0039], [0041], [0050]; and figure 3.	1-4,6-7,9-13,15
Y		5,8,14
Y	US 6430672 B1 (DHONG, Sang Hoo et al.) 06 August 2002 See column 1, lines 51-53; column 4, lines 61-64; and figure 4.	5,8,14
A	US 2009-0150644 A1 (KWON, Young Su et al.) 11 June 2009 See paragraphs [0040]-[0041]; and figure 4.	1-15
A	US 2014-0047197 A1 (KOKRADY, Aman A. et al.) 13 February 2014 See paragraph [0006]; and figure 2.	1-15
A	WO 2011-075170 A1 (MEMOIR SYSTEMS, INC.) 23 June 2011 See paragraph [0104]; and figure 3A.	1-15

☐ Further documents are listed in the continuation of Box C. ☒ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
30 MARCH 2016 (30.03.2016)	**31 MARCH 2016 (31.03.2016)**
Name and mailing address of the ISA/KR Korean Intellectual Property Office Government Complex-Daejeon, 189 Seonsa-ro, Daejeon 302-701, Republic of Korea Facsimile No. 82-42-472-7140	Authorized officer Telephone No.

Form PCT/ISA/210 (second sheet) (January 2015)

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/KR2015/012317

Patent document cited in search report	Publication date	Patent family member	Publication date
US 2006-0012603 A1	19/01/2006	CN 101014933 A	08/08/2007
		CN 101014933 B	27/07/2011
		JP 05422614 B2	19/02/2014
		JP 2008-507034 A	06/03/2008
		JP 2011-238271 A	24/11/2011
		KR 10-0862124 B1	09/10/2008
		KR 10-2007-0030327 A	15/03/2007
		TW I441021 B	11/06/2014
		US 2008-0109611 A1	08/05/2008
		US 7339592 B2	04/03/2008
		US 7634621 B1	15/12/2009
		US 7834881 B2	16/11/2010
		WO 2006-017135 A2	16/02/2006
		WO 2006-017135 A3	05/10/2006
US 6430672 B1	06/08/2002	JP 03606570 B2	05/01/2005
		JP 2002-073412 A	12/03/2002
US 2009-0150644 A1	11/06/2009	KR 10-0922732 B1	22/10/2009
		KR 10-2009-0060666 A	15/06/2009
		US 7958321 B2	07/06/2011
US 2014-0047197 A1	13/02/2014	CN 104520817 A	15/04/2015
		US 9158683 B2	13/10/2015
		WO 2014-031255 A1	27/02/2014
WO 2011-075170 A1	23/06/2011	CN 102870089 A	09/01/2013
		CN 103314363 A	18/09/2013
		CN 103314378 A	18/09/2013
		JP 05715644 B2	13/05/2015
		JP 2013-513884 A	22/04/2013
		US 2010-0241784 A1	23/09/2010
		US 2011-0022791 A1	27/01/2011
		US 2011-0145513 A1	16/06/2011
		US 2011-0145777 A1	16/06/2011
		US 2011-0167192 A1	07/07/2011
		US 2013-0046953 A1	21/02/2013
		US 2014-0310482 A1	16/10/2014
		US 8266408 B2	11/09/2012
		US 8433880 B2	30/04/2013
		US 8504796 B2	06/08/2013
		US 8589851 B2	19/11/2013
		US 8677072 B2	18/03/2014
		US 8935507 B2	13/01/2015
		WO 2011-075167 A1	23/06/2011
		WO 2012-023986 A1	23/02/2012
		WO 2012-026975 A1	01/03/2012

Form PCT/ISA/210 (patent family annex) (January 2015)